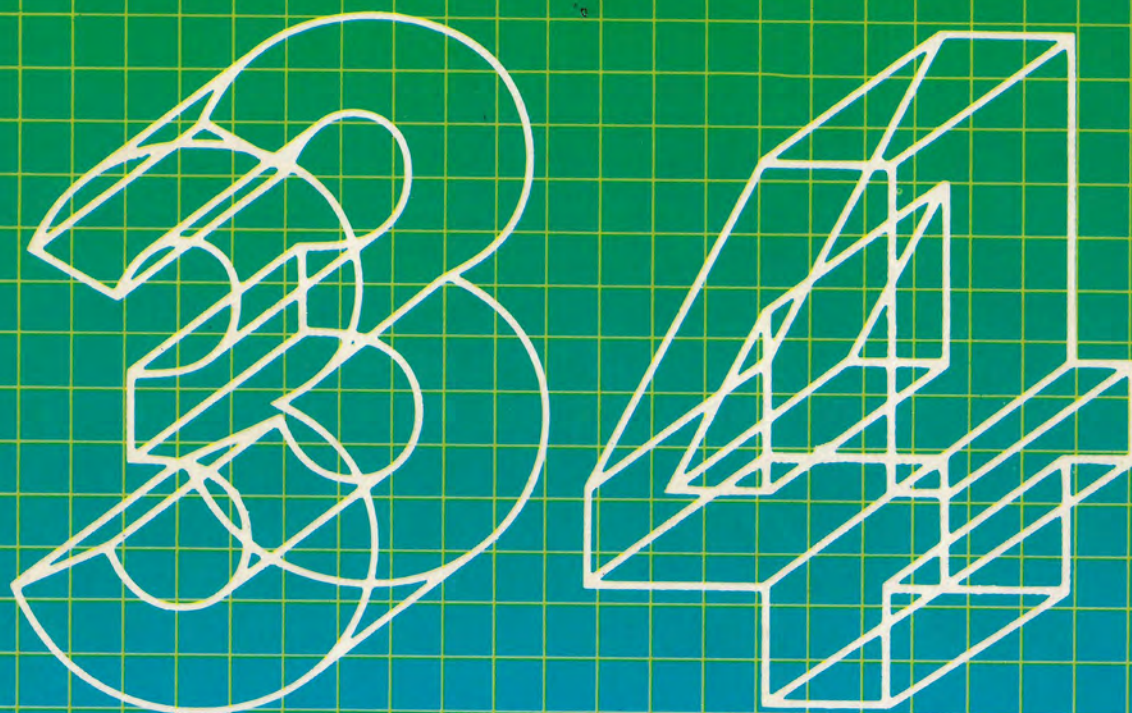


BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



¿UN TALLER DE INFORMATICA?

ATREVASE A HACER BRICOLAGE CON SU O.P.

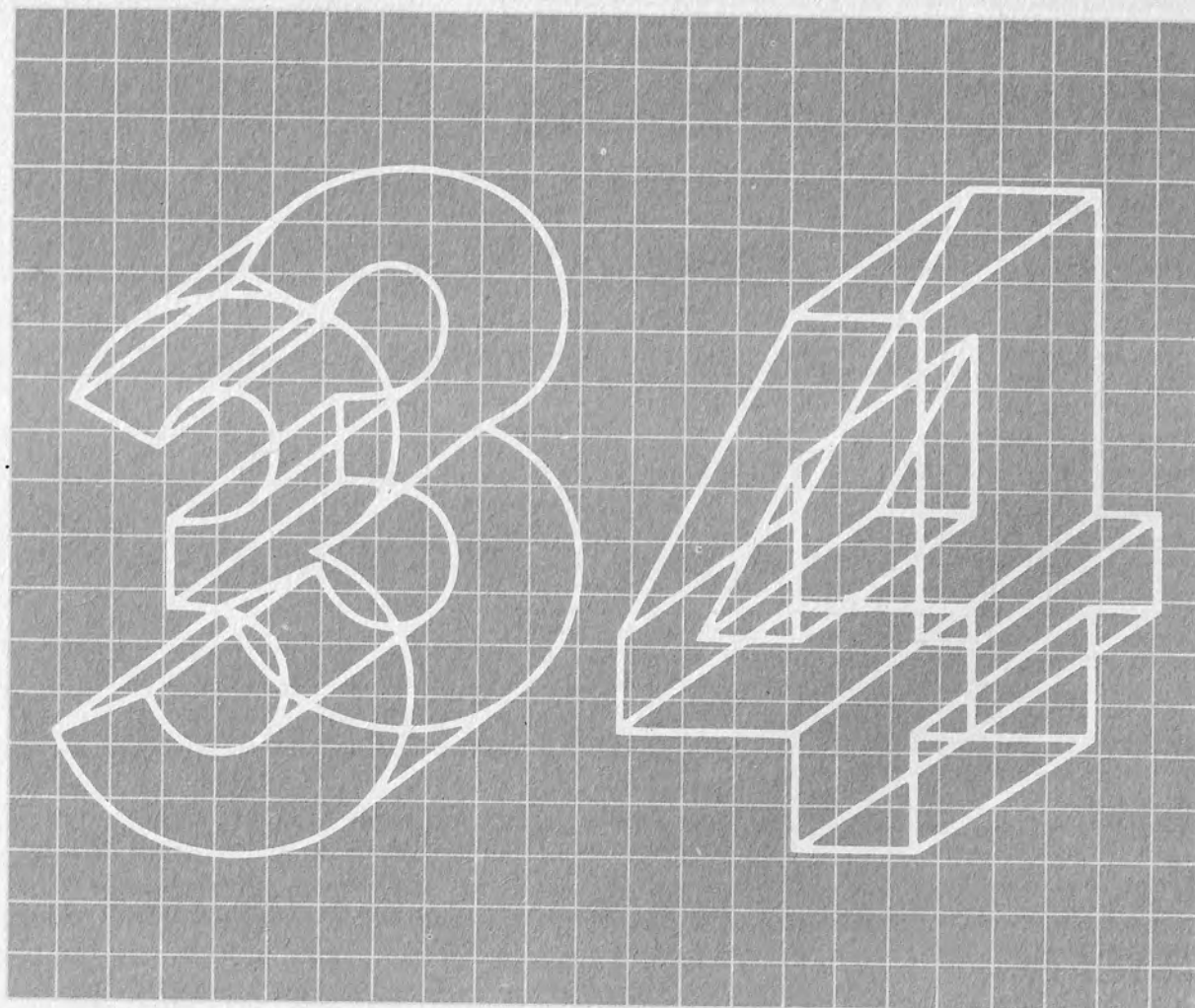
EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:

RICARDO ESPAÑOL CRESPO.

Gerente:

ANTONIO G. CUERPO.

Directora de producción:

MARIA LUISA SUAREZ PEREZ.

Director de la colección:

JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:

BRAVO-LOFISH.

Maquetación:

D. S. M.

Dibujos:

JOSE OCHOA

Tomo 34. «Experiencias y prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubelo, Técnica de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMÁTICA APLICADA: Francisco Blanca, Diplomado en Telecomunicación; Alejandro Marcos, Licenciado en Química. «El Taller del Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:

Pedro Teixeira, 8-2.º. Teléf. 455 09 99. 28020 Madrid.

Publicidad:

Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:

COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.

Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:

CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.

Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-173-8

ISBN de la obra: 84-7688-047-2

Fotocomposición:

ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:

GRAFICAS EMA. (Madrid)

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986

Printed in Spain - Impreso en España.

Suscripciones y números atrasados:

Ediciones Siglo Cultural, S. A.

Pedro Teixeira, 8-2.º. Teléf.: 259 73 31. 28020 Madrid.

Noviembre, 1987

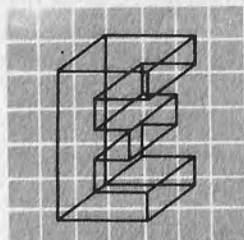
P.V.P. Canarias: 355.—Ptas.

INDICE

■ EXPERIENCIA Y PRACTICAS EN LOGO	5
■ MANEJO DE SPRITES Y ELEMENTOS GRAFICOS	15
■ TRUCOS Y RUTINAS BASICAS	23
■ EL TALLER DEL HARDWARE	29
■ APRENDER CON EL ORDENADOR	37
■ PEQUEÑA HISTORIA DE LA INFORMATICA	47
■ TEMAS MONOGRAFICOS DE VANGUARDIA	51
■ TERMINOLOGIA	55

EXPERIENCIA Y PRACTICAS EN LOGO

Los procedimientos



En este tomo vamos a ver una forma nueva de definir procedimientos.

— DEFINE "nombre
[lista]

Convierte a la lista en la definición de nombre.

nombre: es el nombre que queramos darle al procedimiento.

lista: es la definición, es una lista de listas. El primer elemento de lista será una lista donde se especificarán el nombre de las variables de entrada que ha de tener el procedimiento nombre; el nombre de las variables no debe ir precedido de dos puntos. El segundo dato es una lista que contiene las líneas de definición del procedimiento.

Veamos cómo definimos con DEFINE un procedimiento para dibujar un cuadrado.

```
? DEFINE "A1 [[ ] [ BP BL PONCL 2 REPITE 4 [ AV 40 GD 90 ] ] ]
```

El primer dato de la lista es la lista vacía, el procedimiento A1 no tiene datos de entrada.

Si ejecutamos el procedimiento, en la pantalla se dibujará un cuadrado.

Podemos cargar este procedimiento en el editor.

```
? ED "A1
```

En la pantalla del editor aparecerá la definición de A1.

```
PARA A1  
BP BL PONCL 2 REPITE 4 [ AV 40 GD 90 ]  
FIN
```

Otra forma de definir el procedimiento A1 es descomponiendo en varias líneas la definición.

```
? DEFINE "A1 [[ ] [ BP ] [ BL ]  
[ PONCL 2 ] [ REPITE 4 [ AV 40  
GD 90 ] ] ]
```

Si ahora pasamos el procedimiento A1 al editor obtenemos:

```
? ED "A1  
PARA AD1  
BP  
BL  
PONCL 2  
REPITE 4 [ AV 40 GD 90 ]  
FIN
```

Ya habrás observado que cuando se define un procedimiento a través de la primitiva DEFINE, no se incluye en la lista la palabra FIN.

Para definir el procedimiento A1 con una entrada teclearíamos lo siguiente:

```
? DEFINE "A2 [[ A ] [ BP BL PONCL 2  
REPITE 4 [ AV :A GD 90 ] ] ]
```

Que en la forma acostumbrada es:

```
? IM "A2  
PARA A2 :A  
BP BL PONCL 2 REPITE 4 [ AV :A GD 90 ]  
FIN
```

Veamos otro ejemplo con la primitiva DEFINE.

```
? DEFINE "A3 [[ A B ] [ BP BL PONCL 1  
REPITE 2 [ AV :A GD 90 AV :B GD 90 ] ] ]
```

Hemos definido el procedimiento A3, que dibuja un rectángulo con dos entradas.

Con el Logo podemos crear pequeñas bases de datos.

EXPERIENCIA Y PRACTICAS EN LOGO

Al ejecutar el procedimiento A3 tenemos que indicar dos datos, la base y la altura del rectángulo.

```
? A3 20 60
```

Si queremos ver la definición de un procedimiento definido con DEFINE podemos utilizar la primitiva:

```
TEXTOPROC "nombre
```

que devuelve la definición de nombre en la forma de una lista de listas.

```
? TEXTOPROC "A1  
[ ] [ BP BL PONCL 2 REPITE 4 [ AV 40 GD 90 ] ]
```

```
? TEXTOPROC "A2  
[ A ] [ BP BL PONCL 2 REPITE 4 [ AV :A GD 90 ] ]
```

```
? TEXTOPROC "A3  
[ A B ] [ BP BL PONCL 1 REPITE 2 [ AV :A GD 90 AV :B GD 90 ] ]
```

Esta primitiva nos permite, junto con DEFINE, definir procedimientos utilizando otros ya definidos.

Definimos, por ejemplo, un procedimiento para dibujar una circunferencia:

```
? PARA "CIR  
> BP  
> BL  
> PONCL 3  
> REPITE 36 [ AV 2 GD 10 ]  
> FIN
```

Definimos ahora un nuevo procedimiento que dibuje un cuadrado y una circunferencia; la definición para dibujar esta última figura la aprovechamos del procedimiento CIR.

```
? DEFINE "FIGU PONULTIMO [ REPITE  
4 [ AV 50 GD 90 ] ]  
TEXTOPROC "CIR
```

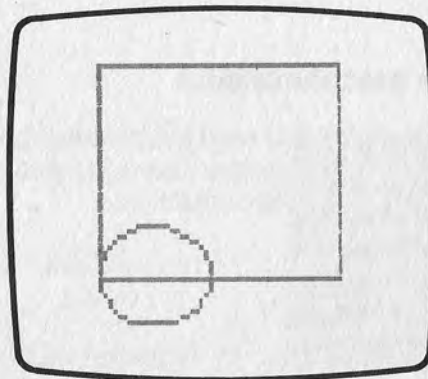
El procedimiento FIGU contiene las siguientes líneas de definición:

```
? IM "FIGU  
PARA FIGU  
BP  
BL  
PONCL 3  
REPITE 36 [ AV 2 GD 10 ]
```

Las listas pueden contener gran cantidad de elementos.

```
REPITE 4 [ AV 50 GD 90 ]  
FIN
```

Al ejecutar FIGU obtenemos el siguiente dibujo:



NOTA: En el Logo del SPECTRUM la sintaxis de esta primitiva es:

```
TEXTO "nombre
```

A los usuarios del AMSTRAD

■ Tratamiento de listas

Vamos a conocer las posibilidades que nos ofrece la versión de Logo del AMSTRAD sobre el manejo de listas.

Esta versión de Logo no diferencia una palabra de una lista y, por tanto, considera a una lista como una sucesión de caracteres. Estos caracteres pueden ser todos, a excepción del carácter en blanco.

Así, pues, las listas no son más que palabras.

Algunos ejemplos de listas podrían ser:

```
— CASA  
— C  
— LACASADEENFRENTE
```

Conozcamos algunas primitivas que posee esta versión para el manejo de listas:

```
— UNE
```

Esta primitiva nos permite unir dos listas:

Por ejemplo, la palabra CASA, podemos formarla mediante dos listas, CA y SA. Para ello introducimos:

? UNE "CA "SA

El Logo nos dará como resultado otra lista formada por las que hemos indicado:

CASA

Observa que las listas que se desean unir tienen que ir precedidas de las comillas:

? UNE CA SA

Nos dará un mensaje de error.

? UNE "DONDE "VAS

Nos dará como resultado:

DONDEVAS

— SIN_PRIMERA

Esta primitiva es equivalente a la que ya hemos visto para otras versiones de Logo:

MENOS PRIMERO (MP)

En la versión del AMSTRAD esta primitiva necesita una variable y una lista para su ejecución.

? SIN_PRIMERA "variable "lista

Si introducimos la siguiente orden:

? SIN_PRIMERA "A "ORDENADOR

en la variable A se almacenará la lista que se especifica sin su primer elemento. Para conocer el resultado, tenemos que ordenar que se escriba el valor de la variable:

? ESCRIBIR :A

y en la pantalla aparecerá:

RDENADOR

La lista con la que se está trabajando puede también estar guardada en una variable:

? HAZ "A PEDRO

? SIN_PRIMERA "V :A

? ESCRIBIR :V

en la pantalla aparecerá:

EDRO

— SIN_ULTIMA

Esta primitiva equivale a la vista para otras versiones de Logo:

MENOS ULTIMO (MU)

Al igual que la primitiva anterior, ésta necesita de una variable y una lista:

? SIN_ULTIMA "V "PARALELO

En la variable V se almacena la lista que se especifica menos su último elemento. Para conocer el resultado introducimos:

? ESCRIBIR :V

y en la pantalla obtendremos:

PARALEL

Al igual que en el caso anterior, la lista puede estar almacenada en una variable.

La siguiente sucesión de órdenes:

? HAZ "A "CAFETERA

? SIN_ULTIMA "V :A

? ESCRIBIR :V

Nos dará como resultado:

CAFETER

Estas son las únicas primitivas de que dispone la versión de Logo del AMSTRAD sobre el manejo de listas.

■ Gestión de la memoria

Conozcamos otras dos primitivas de esta versión de Logo que actúan como comando de gestión de la memoria. Se trata de:

— ESPACIO

Esta primitiva es equivalente a la ya vista:

NODOS

Si introducimos:

? ESPACIO

en la pantalla aparecerá el número de nodos libres que tiene la memoria en ese momento.

— DEJA_ESPACIO

Esta primitiva es equivalente a:

RECICLA

Podemos acceder al elemento que indica una variable.

EXPERIENCIA Y PRACTICAS EN LOGO

Si introducimos:

? DEJA_ESPACIO

al cabo de un pequeño tiempo se habrán liberado de la memoria del ordenador el mayor número de nodos posibles que no fuesen necesarios. De esta manera se habrá aprovechado mucho mejor el espacio de trabajo de la memoria.

■ Cuadro resumen

- UNE "lista1 "lista2 (sólo para el AMS-TRAD).

Une las listas que se especifican formando una única.

- SIN_PRIMERA "variable "lista (sólo para el AMSTRAD)

Asigna a la variable la lista que se especifica menos su primer elemento.

- SIN_ULTIMA "variable "lista (sólo para el AMSTRAD)

Asigna a la variable la lista que se especifica menos su último elemento.

- ESPACIO (sólo para el AMSTRAD)

Nos devuelve el espacio que queda libre en el área de trabajo de la memoria del ordenador.

- DEJA_ESPACIO (sólo para el AMS-TRAD)

Libera el mayor número de nodos de la memoria proporcionando más espacio para trabajar.

- DEFINE "nombre [lista]

Define el procedimiento nombre con las instrucciones contenidas dentro de lista.

- TEXTOPROC "nombre

Devuelve el procedimiento "nombre bajo forma de lista.

■ Ejercicios

1. Este ejercicio servirá para consultar la capital de cualquier país de Europa.

En la ejecución se introducirá el país y se mostrará a continuación su capital.

2. Dibuja una farola.
3. Define los procedimientos necesarios para realizar el siguiente proceso:
 - En la pantalla aparecerá una tabla con dos columnas, una en la que pondrá VOCALES y otra CONSONANTES.
 - Se pedirá que se introduzca una palabra. Habrá que controlar que la palabra no exceda de siete caracteres.
 - Una vez introducida, en la tabla aparecerán las vocales y las consonantes que componen la palabra, cada una de ellas en su columna correspondiente.
 - Aparecerán al pie de la tabla dos mensajes diciéndonos el número de vocales y de consonantes que tiene la palabra.
4. Define los procedimientos necesarios para escribir una palabra que previamente se habrá introducido desde el teclado en:
 - La parte central de la pantalla si se pulsa C.
 - La parte derecha de la pantalla si se pulsa D.
 - La parte izquierda de la pantalla si se pulsa I.
5. ¿Son correctas las siguientes órdenes?
 - ? UNE "CALA "MAR
 - ? HAZ "V "CINE
 - ? SIN_PRIMERA "A ":V
 - ? SIN_ULTIMA "A "GATO
 - ? DEFINE "A [[A ñ AV :A GD 90 AV :A * 80]
 - ? DEFINE "A [[:A :B] [REPITE 2 [AV :A GD 90 AV :B GD 90]]]
 - ? DEFINE "A [[] [SL PONPOS [20 20] BL PONRUMBO 90 AV 60]

■ Solución a los ejercicios

1.

La lista de los países de Europa y de sus capitales respectivas la almacenamos en dos variables en el siguiente procedimiento.

A contiene la lista de países.

Los paréntesis facilitan la claridad de las operaciones.

B contiene la lista de capitales.
Ambas listas se corresponden en orden
(país-capital).

```
? PARA LISTAS
> HAZ "A [ URSS [ R.F.DE.ALEMANIA ]
  ITALIA REINO.UNIDO FRANCIA ES-
  PAÑA POLONIA YUGOSLAVIA RU-
  MANIA R.D.DE.ALEMANIA CHE-
  COSLOVAQUIA HOLANDA HUN-
  GRIA BELGICA PORTUGAL BULGA-
  RIA GRECIA AUSTRIA SUIZA SUE-
  CIA DINAMARCA FINLANDIA NO-
  RUEGA IRLANDA ALBANIA
  LUXEMBURGO MALTA ISLANDIA ]
> HAZ "B [ MOSCU BONN ROMA LON-
  DRES PARIS MADRID VARSOVIA
  BELGRADO BUCAREST BERLIN
  PRAGA AMSTERDAM BUDAPEST
  BRUSELAS LISBOA SOFIA ATENAS
  VIENA BERNA ESTOCOLMO CO-
  PENHAGUE HELSINKI OSLO DU-
  BLIN TIRANA LUXEMBURGO
  LA.VALLETТА REYKIAVIK ]
> HAZ "I 1
> FIN
```

El procedimiento P1 presenta la panta-
lla siguiente:

PAIS	CAPITAL

INTRODUCE EL PAIS

```
? PARA P1
> PONCURSOR [ 9 5 ]
> ES "PAIS
> PONCURSOR [ 23 5 ]
> ES "CAPITAL
> SL
```

```
> PONPOS [ -130 40 ]
> BL
> REPITE 2 [ AV 40 GD 90 AV 260 GD
  90 ]
> AV 20 GD 90
> AV 260 RE 130
> GI 90 AV 20 RE 40
> PONCURSOR [ 10 12 ]
> ES [ INTRODUCE EL PAIS ]
> FIN
```

P2 lo definimos para leer del teclado el
país que elijamos de la lista de países de Euro-
pa.

Al introducir el país ten en cuenta que
debes especificarlo exactamente como apare-
ce en la lista de países.

```
? PARA P2
> PONCURSOR [ 4 7 ]
> HAZ "L LL
> FIN
```

En P3 localizamos la capital del país in-
troducido.

```
? PARA P3
> HAZ "P (ELEMENTO :I :A)
> HAZ "P LISTA :P
> SI :P = :L [ PONCURSOR [ 23 7 ] ES
  ELEMENTO :I :B ]
  [ HAZ "I :I + 1 P3 ]
> FIN
```

En P4 preguntamos si se quiere o no in-
troducir otro país.

```
? PARA P4
> PONCURSOR [ 10 12 ]
> ES [ ¿OTRO PAÍS? ( S / N ) ]
> HAZ "T LC
> SI :T = "S [ TO ]
> SI :T = "N [ ALTO ]
> P4
> FIN
```

Finalmente definimos el procedimiento
TO, que inicializa la pantalla y llama a los pro-
cedimientos definidos anteriormente.

```
? PARA TO
> BP
> BT
> OT
> LISTAS
> P1
```

***Puedes hacer procedimientos recursivos
con listas.***

EXPERIENCIA Y PRACTICAS EN LOGO

- > P2
- > P3
- > P4
- > FIN

Empecemos a introducir países.

? TO

Y consultemos, por ejemplo, la capital del REINO.UNIDO tecleándolo tal y como aparece en la lista.

En un instante aparecerá debajo de la columna CAPITALES el nombre de la capital (LONDRES).



2.

Definimos un procedimiento para inicializar la pantalla y para llamar al procedimiento FAROLA.

- ? PARA D
- > BP
- > OT
- > FAROLA
- > FIN

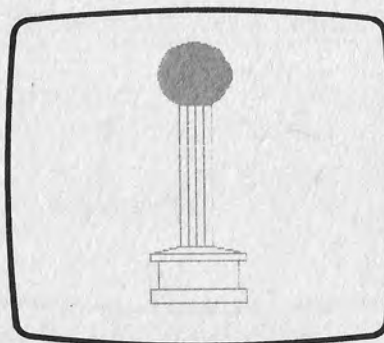
El siguiente procedimiento es el que definimos para dibujar la farola:

- ? PARA FAROLA
- > SL
- > PONPOS [-40 -70]

- > BL
- > PONRUMBO 90
- > REPITE 2 [AV 60 GD 90 AV 10 GD 90]
- > AV 5 GI 90
- > REPITE 2 [AV 20 GD 90 AV 50 GD 90] AV 20
- > GI 90 AV 5
- > GD 90
- > REPITE 2 [AV 5 GD 90 AV 60 GD 90]
- > AV 5
- > PONRUMBO 75
- > REPITE 2 [AV 20 GD 15]
- > AV 20
- > SL
- > PONPOS [-21 -40]
- > PONRUMBO 0
- > BL
- > REPITE 4 [REPITE 2 [AV 100 GD 90 AV 5 GD 90] GD 90 AV 5 GI 90]
- > SL
- > PONPONS [-21 60]
- > BL
- > PONRUMBO 300
- > REPITE 31 [AV 4 GD 10]
- > SL
- > PONPONS [-10 70]
- > BL
- > RELLENA
- > FIN

Lo ejecutamos tecleando:

? D



Descubre otra forma de definir procedimientos con la primitiva DEFINE.
Si definimos un procedimiento con la primitiva DEFINE, no incluiremos la palabra FIN.

Definimos el procedimiento P1, que es el que va a inicializar el proceso y el que va a dibujar en la pantalla la tabla en donde van a aparecer las vocales y las consonantes:

```
? PARA P1
> BP
> BT
> OT
> SL
> PONPONS [ -80 12 ]
> BL
> REPITE 2 [ AV 100 GD 90 AV 170 GD
  90 ]
> AV 80 GD 90
> AV 170 RE 102
> GI 90 AV 20
> RE 100
> PONCURSOR [ 11 2 ]
> ES "VOCALES
> PONCURSOR [ 19 2 ]
> ES "CONSONANTES
> P2
> FIN
```

Este procedimiento realiza al final una llamada a P2:

```
? PARA P2
> PONCURSOR [ 2 15 ]
> ES [ INTRODUCE UNA PALABRA: ]
> PONCURSOR [ 25 15 ]
> ES ".....
> PONCURSOR [ 32 15 ]
> PONCT 0
> ES ".....
> PONCT 7
> PONCURSOR [ 25 15 ]
> HAZ "P LL
> HAZ "P UL :P
> HAZ "C CUENTA :P
> SI :C > 7 [ P2 ]
> P3
> FIN
```

Este procedimiento se encarga de escribir el mensaje de introducción de la palabra, de pedir que se introduzca y de asignársela a la variable P:

Una vez que tenemos la palabra (que se

ha introducido mediante una lectura de lista), la transformamos en una palabra Logo.

A continuación contamos los elementos que tiene la palabra (los caracteres) para controlar que no excedan de siete. El número de elementos que tiene los guardamos en la variable C.

Si el número de elementos excede de siete, volvemos a ejecutar el procedimiento P2 y se pedirá otra palabra. Si no excede, ejecutamos el procedimiento P3:

```
? PARA P3
> HAZ "I 1
> HAZ "CV 0
> HAZ "CC 0
> REPITE :C [ P4 ]
> P5
> FIN
```

P3 se encarga de inicializar las siguientes variables:

— Variable I: Es la encargada de llevar el contador elementos de la palabra y a su vez la que va a servir para indicar la fila en donde se tienen que escribir bien la vocal o bien la consonante.

— Variable CV: Es la que va a contar el número de vocales que tiene la palabra.

— Variable CC: Es la que va a contar el número de consonantes que tiene la palabra.

Una vez inicializadas las variables, ordenamos que se repita el procedimiento P4 tantas veces como elementos tiene la palabra:

```
? PARA P4
> HAZ "E ELEMENTO :I :P
> SI :E = "A [ VOCAL ALTO ]
> SI :E = "E [ VOCAL ALTO ]
> SI :E = "I [ VOCAL ALTO ]
> SI :E = "O [ VOCAL ALTO ]
> SI :E = "U [ VOCAL ALTO ]
> CONSONANTE
> FIN
```

P4 va asignado a la variable E el elemento que indica I. A continuación se realizan cinco preguntas para saber si el elemento de la palabra es igual a alguna de las cinco vocales. De serlo, se ejecuta el procedimiento VOCAL y cuando regresa, detenemos la ejecu-

Si un procedimiento no contiene ninguna entrada, el primer dato de la lista que lo define es la lista vacía.

EXPERIENCIA Y PRACTICAS EN LOGO

ción devolviendo el control al REPITE. De no serlo, significará que a la fuerza el elemento tiene que ser una consonante, por lo que ejecutamos el procedimiento CONSONANTE:

```
? PARA VOCAL
> PONCURSOR LISTA 14 3 + :I
> ES :E
> HAZ "I :I + 1
> HAZ "CV :CV + 1
> FIN
```

```
? PARA CONSONANTE
> PONCURSOR LISTA 24 3 + :I
> ES :E
> HAZ "I :I + 1
> HAZ "CC :CC + 1
> FIN
```

En ambos procedimientos, situamos el cursor en una columna determinada y en la fila que resulta de la operación, 3 + :I. Una vez situados, escribimos la variable E, que es la que contiene el elemento que se está tratando, incrementamos en uno el contador I y las variables que cuentan las vocales o las consonantes, respectivamente.

Cuando se termina de realizar el REPITE en el procedimiento P3, se hace una llamada a P5:

```
? PARA P5
> PONCURSOR [ 2 17 ]
> ES FRASE [ NUMERO DE VOCAL-
ES ] :CV
> PONCURSOR [ 2 19 ]
> ES FRASE [ NUMERO DE CONSO-
NANTES ] :CC
> FIN
```

P5 es el que se encarga de escribir el número de vocales y consonantes que tiene la palabra que hemos introducido.

Con todos estos procedimientos, podemos ejecutar el proceso. Para ello, introducimos:

```
? P1
```

y en la pantalla aparece:

VOCALES	CONSONANTES

INTRODUCE UNA PALABRA: ■.....

Si, por ejemplo, introducimos la palabra:

VENTANA

obtenemos:

VOCALES	CONSONANTES
E	U
A	N
A	N

INTRODUCE UNA PALABRA: VENTANA
NUMERO DE VOCALES 3
NUMERO DE CONSONANTES 4

4.

El procedimiento A es el que hace la petición de la palabra y nos presenta el menú de opciones:

```
? PARA A
> BP
> BT
> OT
> PONCURSOR [ 2 2 ]
> ES [ INTRODUCE UNA PALABRA: ]
```

Los datos de entrada en un procedimiento definido con DEFINE no llevan los dos puntos delante.

```

> PONCURSOR [ 25 2 ]
> HAZ "P LL
> HAZ "P ULTIMO :P
> PONCURSOR [ 2 8 ]
> ES [ ¿DONDE QUIERES ESCRIBIR LA
  PALABRA? ]
> PONCURSOR [ 13 12 ]
> ES [ C:CENTRO ]
> PONCURSOR [ 13 14 ]
> ES [ D: DERECHA ]
> PONCURSOR [ 13 16 ]
> ES [ I: IZQUIERDA ]
> B
> FIN

```

El B es el que solicita que se elija una opción y controla que sólo se pulse C, D o I:

```

? PARA B
> PONCURSOR [ 11 20 ]
> ES [ ELIGE OPCION: ]
> HAZ "O LC
> PONCURSOR [ 24 20 ]
> ES :O
> SI Y ( Y ( NO :O = "C )
  ( NO :O = "D ) )
  (( NO :O = "I ) )
  [ B ] [ C ]
> FIN

```

El procedimiento C escribe la palabra en la posición elegida, para lo cual, lo único que tiene que calcular es la columna en donde tiene que empezar a escribirse la palabra introducida, dependiendo del número de elementos que tenga:

```

? PARA C
> BP
> BT
> HAZ "C CUENTA :P
> SI :O = "C [ PONCURSOR
  LISTA REDONDEA ( 19 -
  (:C / 2 ) ) 2 ES :P ]
> SI :O = "I [ PONCURSOR [ 0 2 ] ES :P ]
> SI :O = "D [ PONCURSOR LISTA 39 -
  :C 2 ES :P ]
> FIN

```

Para ejecutarlo bastará con introducir:

```
? A
```

5.

```
— ? UNE "CALA "MAR
CORRECTO
```

Las listas CALA y MAR se unen formando una sola: CALAMAR.

```
— ? HAZ "V "CINE
? SIN_PRIMERA "A ":V
INCORRECTO
```

No se pueden especificar las comillas si queremos utilizar el contenido de una variable.

```
— ? SIN_ULTIMA "A "GATO
CORRECTO
```

La variable A contendrá la lista: GAT.

```
— ? DEFINE "A [ [ A ] AV :A GD 90 AV
  :A * 80 ]
INCORRECTO
```

Faltan los corchetes que para encerrar las instrucciones que definen el procedimiento.

LOGO devuelve el mensaje de error:

```
DEFINE NO ACEPTA [ AV :A GD () AV
:A * 80 ] COMO DATO.
```

```
— ? DEFINE "A [ [ :A :B ] [ REPITE
  2 [ AV :A GD 90 AV :B GD 90 ] ] ]
INCORRECTO
```

Las variables de entrada que contiene la primera lista de la lista de definición no deben llevar los dos puntos delante.

Sería correcto:

```
? DEFINE "A [ [ A B ] [ REPITE 2 [ AV:A
  GD 90 AV :B GD 90 ] ] ]
```

```
— ? DEFINE "A [ [ ] [ SL PONPOS [ 20
  20 ]
  BL PONRUMBO 90 AV 60 FIN ] ]
INCORRECTO
```

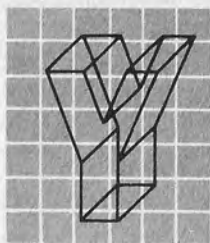
Cuando ejecutemos el procedimiento A, el Logo mostrará el mensaje:

```
NO SE COMO HACER PARA FIN
```

Con la primitiva TEXTOPROC pueden escribirse procedimientos que controlan y manipulan otros procedimientos.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

Alta resolución en el ZX-SPECTRUM



A vimos que la pantalla del SPECTRUM estaba compuesta de 22 líneas de 32 caracteres más dos líneas inferiores también con 32 caracteres. Estos caracteres por línea van numerados del 0 al 31 y las líneas del 0 al 21.

Pero todo esto es baja resolución o más bien modo carácter. Para trabajar en alta resolución en el SPECTRUM no hay que hacer nada. No hay que cambiar el modo de pantalla. El mismo modo de texto nos sirve para alta resolución. Esto nos permite mezclar textos y gráficos con una facilidad que sólo se iguala a la del IBM.

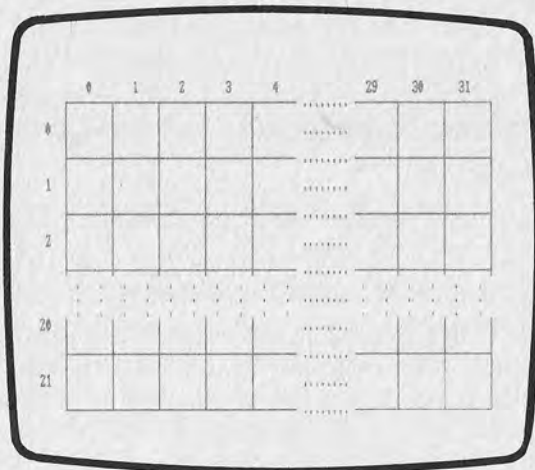


Fig. 1. Organización de las filas y columnas de caracteres en el SPECTRUM.

Como tenemos 32 caracteres por línea y cada carácter se compone de una matriz de 8×8 puntos, esto nos dice que tenemos 256 puntos de ancho (32×8) y, como tenemos 22 líneas de 8 puntos de altura cada una de ellas, 176 puntos de alto.

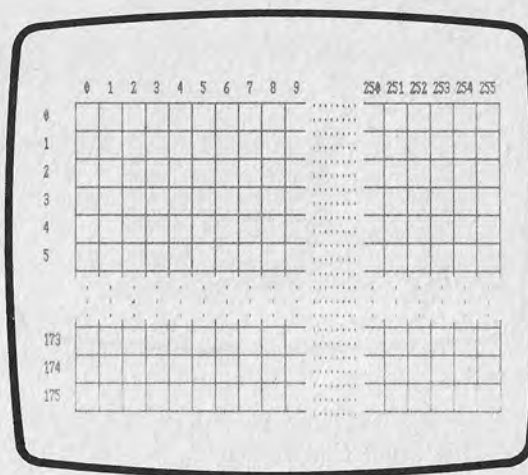


Fig. 2. Formación en la alta resolución en el SPECTRUM.

Estos puntos van a ser direccionables uno a uno desde programa y podremos hacer que estén iluminados o apagados a voluntad. Al igual que hacíamos con la pantalla, es necesario dar un sistema de referencia, un sistema de coordenadas, para decirle al ordenador qué punto es el que queremos tratar. Por ello, para todas las operaciones, tomaremos como origen de coordenadas la esquina superior izquierda de la pantalla. Según esto, las coordenadas en X aumentarán hacia la derecha y las coordenadas en Y hacia abajo.

Aunque podemos escribir mensajes en las dos líneas reservadas al sistema (líneas 23 y 24) y aunque algunos programas comerciales en código máquina puedan dibujar gráficos en estas dos líneas, nosotros, desde el BASIC, no podemos dibujar en ellas. Cualquier intento por nuestra parte de hacerlo nos dará un mensaje de error.

Hay que decir que podemos dar color a los puntos que dibujemos y al fondo sobre el que lo hagamos. El único problema que aparece es que cada grupo de 8×8 puntos (un ca-

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

rácter) sólo puede tener un color de papel y un color de tinta. No podemos tener dos puntos con colores distintos si ambos están situados en la misma posición de carácter. En la práctica esto no llega a ser muy importante, pues una buena planificación de la pantalla puede llevarnos a conseguir dibujos en los que cada color se encuentre en su sitio y no se mezcle con los demás colores.

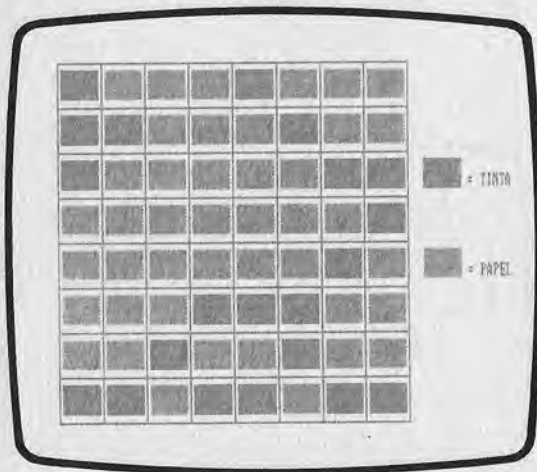


Fig. 3.

Como podemos ver en la figura 3, los puntos en negro corresponden al color de la tinta y los grises al color del papel. Esta matriz de 8 x 8 puntos podría pertenecer a un dibujo que ocupase toda la pantalla. Al estar todos los puntos dentro de los límites de un carácter, todos han de tener el mismo color.

La pantalla del SPECTRUM se encuentra almacenada a partir de la posición de memoria 16384. Esto significa que podemos grabar la pantalla como si de Bytes se tratase para, más tarde, poderla recuperar y para que actúe como pantalla de presentación, o cualquier otra cosa, de nuestros programas. Para ello, podemos utilizar el comando SAVE en el formato de grabat Bytes. Este es:

SAVE "nombre" CODE Dir.inicio,No.de BYTES
por lo que para grabar una pantalla tenemos que poner:

SAVE "Pantalla" CODE 16384,6144

Al hacer esto no hemos grabado el color de la pantalla. Sólo hemos almacenado los puntos de la pantalla, encendidos o apagados. Esto es debido a que tras la zona donde está almacenada la pantalla, tenemos otra zona donde se encuentran los colores. Cada posición de memoria dentro de esta zona almace-

na el color de un carácter según el siguiente formato:

$$\text{COLOR} = \text{TINTA} + 8 * \text{PAPEL} + 64 * \text{BRILLO} + 128 + \text{FLASH}$$

lo cual es como decir que los tres primeros Bytes me dan el color de la tinta, los tres siguientes el color del papel, el bit número 6 si el carácter tiene brillo (1) o no (0) y el último si está parpadeando (1) o no (0).

Esta zona, llamada de atributos, ocupa desde la dirección de memoria 22528 hasta la 23296, lo cual nos da un total de 768 Bytes. Si multiplicamos el número de líneas (contando las dos de abajo) por el número de caracteres por línea, podemos ver que nos sale el mismo número. 32 caracteres X 24 líneas = 768.

Según todo lo visto, podemos grabar toda la pantalla, incluida la zona de atributos con sólo poner:

SAVE "Pantalla" CODE 16384,6912

pero esto tampoco es necesario, pues el Basic del SPECTRUM nos permite grabar la pantalla sin necesidad de saber dónde se encuentra. Sólo hay que escribir:

SAVE "Pantalla" SCREEN\$

pero es interesante saber dónde se encuentra por si queremos grabar sólo los puntos de la pantalla o los atributos.

SAVE "Atributos" CODE 22528,768

Una vez grabada la pantalla en cinta de casete, para recuperarla sólo tenemos que poner:

LOAD "Nombre de la pantalla" CODE

o mejor aún:

LOAD "Nombre de la pantalla" SCREEN

Los comandos de la alta resolución

Una vez entendido cómo es y dónde se almacena la pantalla en nuestro ordenador, vamos a ver cómo podemos dibujar en él. De momento ya sabemos cómo imprimir. Sólo tenemos que utilizar el comando PRINT (seguido de AT coorY, coorX si queremos que sea en cierta posición). Esto significa que, al contrario que en otros ordenadores como el MSX o el AMSTRAD, no podemos escribir texto en posiciones de la pantalla que no sean de carácter. Siempre tendremos que colocar las letras y los demás signos de que disponemos en posiciones de carácter. Nunca entre caracteres.

LA INSTRUCCION PLOT

Con esta instrucción podemos iluminar un punto que se encuentre en la pantalla. La sintaxis es la siguiente:

PLOT X,Y

donde X e Y son las coordenadas de dicho punto en la pantalla. El color de dicho punto vendrá dado por la última instrucción INK utilizada en el programa, por ello, si queremos dibujar un punto en la posición de pantalla 100,100 de color amarillo, podemos poner:

INK 6:PLOT 100,100

Tenemos otra manera de hacer esto. Basta con introducir la propia instrucción de color tras la palabra PLOT y antes de las coordenadas.

PLOT INK 6;0,0

Pero no sólo podemos utilizar el comando INK, sino todos los demás que vimos cuando hablamos del color.

PLOT INK n;X,Y
PLOT PAPER n;X,Y
PLOT FLASH n;X,Y
PLOT BRIGHT n;X,Y
PLOT INVERSE n;X,Y
PLOT OVER n;X,Y

y no sólo podemos incluir un comando, sino todos los que queramos mientras estén separados por comas.

PLOT OVER 1;INK 3;BRIGHT 0;100,100

Si incluimos el comando PAPER, el ordenador no sólo iluminará, o apagará el punto al que nos referimos, sino que además pondrá el color del papel como le hayamos indicado. Si lo que estamos utilizando es OVER, entonces, el SPECTRUM encontrará el punto al que nos referimos en caso de que esté apagado y lo apagará si estuviese encendido.

Para entender mejor como actúa esta instrucción, vamos a ver un programa que nos llena de puntos la pantalla. El resultado se nos aparece como una noche estrellada (si las estrellas fuesen de colores).

```
1000 REM *****
1010 REM * NOCHE ESTRELLADA *
1020 REM *****
1030 REM
1040 REM *****
1050 REM * PROGRAMA DEMOSTRACION PARA VER EL *
1060 REM * USO DEL PLOT EN EL SPECTRUM. *
1070 REM *****
1080 REM
1090 PAPER 0
1100 INK 6
1110 BORDER 0
1120 FLASH 0
1130 BRIGHT 0
1140 CLS
1150 DIM X(500)
1160 DIM Y(500)
1170 REM
1180 REM *** DIBUJO DEL CIELO ESTRELLADO ***
1190 REM
1200 PRINT#0;PAPER 1;INK 7;"DIBUJANDO EL CIELO";
1210 RANDOMIZE 0
1220 FOR I=1 TO 500
1230 LET X=INT(RND*256)
1240 LET Y=INT(RND*176)
1250 LET C=INT(RND*7+1)
1260 LET F=INT(RND*2)
1270 LET B=INT(RND*2)
1280 PLOT PAPER 0;INK C;BRIGHT B;FLASH F;X,Y
1290 LET X%(I)=X
1300 LET Y%(I)=Y
1310 NEXT I
1320 REM
1330 REM *** BUCLE DE RETARDO ***
1340 REM
1350 FOR I=1 TO 1500
1360 NEXT I
1370 REM
1380 REM *** BORRADO DEL CIELO ESTRELLADO ***
```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```

1390 REM
1400 BEEP .1,30
1410 BEEP .1,20
1420 BEEP .1,10
1430 BEEP .2,0
1440 PRINT #0;" "; : REM 32 ESPACIOS
1450 PRINT #0; AT 1,0:PAPER 1;INK 7;" BORRANDO EL CIELO ";
1460 FOR I=1 TO 500
1470   LET X=X%(I)
1480   LET Y=Y%(I)
1490   PLOT PAPER 1;FLASH 0;BRIGHT 0;OVER 1;X,Y
1500 NEXT I
1510 STOP

```

La figura 4 nos muestra cómo nos puede quedar la pantalla tras la ejecución.



La instrucción DRAW

Con esta segunda instrucción podremos dibujar líneas y arcos de circunferencia. Actúa mediante coordenadas relativas. Esto es, no le damos las coordenadas absolutas de la pantalla a las que queremos que trace una línea, sino las coordenadas relativas al último punto trazado o iluminado. Su sintaxis es:

DRAW X,Y,R

donde X e Y son las coordenadas (relativas al último punto) donde queremos llegar y R es la longitud del arco de circunferencia que queremos que trace mientras dibuja la línea. Normalmente, este tercer parámetro no se suele utilizar a no ser que deseemos dibujar arcos; por ello, podemos prescindir de él y no incluirlo si no lo necesitamos.

Cuando decimos que la sentencia DRAW dibuja tomando las coordenadas que se le dan como relativas al último punto trazado, queremos decir que si, por ejemplo, hacemos un PLOT en 20,20 y queremos dibujar una línea recta desde esta posición hasta la 100,100, tendremos que escribir:

PLOT 20,20:DRAW 80,80

si quisiésemos que de paso nos dibujase un arco de circunferencia podríamos escribir:

PLOT 20,20:DRAW 80,80,3.14

El siguiente programa nos muestra algunas de las cosas que podemos hacer combinando DRAW con PLOT.

```

1000 REM *****
1010 REM * NEW YORK *
1020 REM *****
1030 REM
1040 REM *****
1050 REM * PROGRAMA DEMOSTRACION PARA VER EL USO DEL *
1060 REM * COMANDO DRAW EN EL SPECTRUM. *
1070 REM *****
1080 REM
1090 REM *** INICIALIZACION Y DIBUJO DE LA PANTALLA ***
1100 REM
1110 PAPER 0
1120 INK 6
1130 BRIGHT 1
1140 FLASH 0
1150 OVER 0
1160 CLS

```

```

1170 PRINT #0;AT 1,0:PAPER 1;INK 7;" VISTA AEREA DE NEW YORK ";
1180 REM
1190 PLOT 0,0 : REM SE COLOCA EL CURSOR GRAFICO EN 0,0
1200 DRAW 255,0 : REM DIBUJAMOS UNA LINEA HACIA LA DERECHA
1210 DRAW 0,130 : REM DIBUJAMOS UNA LINEA HACIA ABAJO
1220 DRAW -255,0 : REM DIBUJAMOS UNA LINEA HACIA LA IZQUIERDA
1230 DRAW 0,-130 : REM DIBUJAMOS UNA LINEA HACIA ARRIBA
1240 REM
1250 LET XMIN=10
1260 LET YMIN=10
1270 LET XMAX=245
1280 REM
1290 REM *** RECOGIDA DE OPCION ***
1300 REM
1310 RANDOMIZE 0
1320 PRINT AT 18,0;
1330 PRINT "COMO QUIERES LOS EDIFICIOS?"
1340 PRINT INVERSE 1;"L";INVERSE 0;"LENOS O ";
1350 PRINT INVERSE 1;"V";INVERSE 0;"ACIOS"
1360 LET A$=INKEY$
1370 IF A$="" THEN GOTO 1360
1380 IF A$="L" OR A$="l" THEN LET SW=1:GOTO 1410
1390 IF A$="V" OR A$="v" THEN LET SW=0:GOTO 1410
1400 GOTO 1360
1410 PRINT AT 18,0;
1420 FOR I=1 TO 2
1430 PRINT " " : REM 32 ESPACIOS.
1440 NEXT I
1450 PRINT AT 18,0;
1460 PRINT FLASH 1;"PULSA 'P' PARA PARAR"
1470 REM
1480 REM *** DIBUJO DE NEW YORK ***
1490 REM
1500 LET X1=INT(RND*XMAX+XMIN)
1510 LET Y1=INT(RND*100+YMIN)
1520 LET X2=INT(RND*XMAX+XMIN)
1530 LET Y2=120
1540 LET C=INT(RND*7+1)
1550 IF ABS(X2-X1)>80 THEN GOTO 1500
1560 LET X2=ABS(X2-X1)
1570 LET Y2=ABS(Y2-Y1)
1580 INK C
1590 IF SW=0 THEN PLOT X1,Y1:DRAW X2,0:DRAW 0,Y2:DRAW -X2,0:DRAW 0,-Y2
1600 IF SW=1 THEN GOSUB 1700
1610 REM
1620 REM *** LECTURA DEL TECLADO Y RETARDO ***
1630 REM
1640 FOR I=1 TO 100
1650 IF INKEY$="P" THEN GOTO 1680
1660 NEXT I
1670 GOTO 1500
1680 CLS
1690 STOP
1700 REM
1710 REM *****
1720 REM * DIBUJO DE RECTANGULOS RELLENOS *
1730 REM *****
1740 REM
1750 A=X2
1760 IF X2<Y2 THEN LET A=Y2
1770 FOR X=0 TO INT(A/2)-1
1780 PLOT X1+X,Y1+X
1790 DRAW X2-X,0
1800 DRAW 0,Y2-X
1810 DRAW -X2+X,0
1820 DRAW 0,-Y2+X
1830 NEXT X
1840 RETURN

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

Como curiosidad para ver las cosas que se pueden hacer utilizando el tercer argumento de la sentencia DRAW, os proponemos introducir las siguientes líneas. A ver qué pasa:

```
10 PLOT 140,20
20 DRAW OVER 1;55,55,51^4
30 PLOT 130,65
40 DRAW 20,20,51 4
```

Si variamos el tercer argumento de la sentencia DRAW que hay en las líneas 20 y 40, podremos variar el efecto. Un rato jugando con este pequeño programa nos hará entender mejor el funcionamiento de este argumento.

```
10 FOR I = 0 TO 40
20 PLOT 128,88
30 DRAW 5,5,I
40 NEXT I
```

La instrucción CIRCLE

También podemos dibujar círculos con el SPECTRUM. Lo único que hay que hacer es utilizar el comando CIRCLE y decirle las coordenadas del centro y el radio que queremos que tenga. La sintaxis de esta instrucción es:

CIRCLE X,Y,R

donde X e Y son las coordenadas del centro del círculo y R es el radio.

En el caso de que el círculo no quepa en la pantalla, por haber dado un radio demasiado grande, el ordenador nos mostrará un mensaje de error.

El pequeño programa que aparece a continuación nos da una idea de las cosas que podemos hacer con el comando CIRCLE aparte de poder hacer círculos normales y corrientes:

```
10 FOR I = 40 TO 120 STEP 10
20 FOR J = 50 TO 200 STEP 10
30 CIRCLE J,I,20
40 NEXT J
50 NEXT I
```

La instrucción POINT

Muchas veces, cuando estamos realizando un programa en el que hay figuras móviles por la pantalla, es necesario saber si un punto está encendido o apagado. Esto pode-

mos saberlo con ayuda de la sentencia POINT. Su sintaxis es:

POINT (X,Y)

donde X e Y son las coordenadas del punto que queremos testear. Como el resultado de este comando es un número (0 si está apagado y 1 si está encendido), habrá que asignar su valor a una variable.

LET PUNTO = POINT (X,Y)

El comando ATTR

Al igual que con POINT podemos saber si un punto está iluminado o apagado, con ATTR podemos saber si una cierta posición de carácter tiene un color u otro. Este comando nos dirá el color de la posición de carácter que se encuentra en la columna X de la fila Y. Su sintaxis es:

LET COLOR = ATTR (X,Y)

El resultado de este comando es un número entre 255 que resulta de:

COLOR = TINTA + 8 * PAPEL + 64 * BRILLO + 128 * FLASH

lo que como se recordará es la forma en que el ordenador almacena el color de la pantalla en la zona de atributos.

Rellenado de figuras

Al contrario que muchos ordenadores, el SPECTRUM no dispone de ninguna sentencia FILL o PAINT que nos rellene de color figuras cerradas. A continuación vamos a dar una rutina que, aunque un poco lenta, nos rellena cualquier figura con sólo decirle un punto dentro de la figura a rellenar y el color en que queremos que lo haga. (Véase el programa de la página 21.)

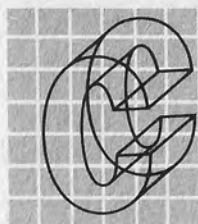
Puede que algunas veces sea necesario llamar dos o más veces a la rutina. Esto ocurre cuando es una figura muy irregular. En estos casos la rutina puede dejar alguna parte del dibujo sin colorear. Por ello habrá que volver a llamar a la rutina, cambiando las coordenadas de forma que queden dentro de la parte sin pintar.

```

9500 REM *****
9510 REM *
9520 REM *  RUTINA  PAINT  PARA  SPECTRUM  *
9530 REM *  =====  *
9540 REM *  *
9550 REM *****
9560 REM
9570 REM *****
9580 REM *
9590 REM *  VARIABLES QUE HAY QUE PASARLE A LA RUTINA  *
9600 REM *  -----  *
9610 REM *  *
9620 REM *  XO = COORDENADA X DENTRO DE LA FIGURA.  *
9630 REM *  YO = COORDENADA Y DENTRO DE LA FIGURA.  *
9640 REM *  *
9650 REM *  C = COLOR DE LA TINTA.  *
9660 REM *  *
9670 REM *****
9680 REM
9690 LET D=0
9700 LET VY=1
9710 LET Y=YO
9720 LET C=0
9730 LET I=0
9740 LET VX=1
9750 LET X=XO
9760 IF POINT (X,Y)=1 THEN GOTO 9890
9770 PLOT INK C;X,Y
9780 LET X=X+VX
9790 LET C=C+VX
9800 IF POINT (X,Y)=0 THEN GOTO 9770
9810 IF I=1 THEN GOTO 9860
9820 LET X=XO-1
9830 LET VX=-1
9840 LET I=1
9850 GOTO 9790
9860 LET XO=XO+C/2
9870 LET Y=Y+VY
9880 GOTO 9720
9890 IF D=1 THEN RETURN
9900 LET VY=-1
9910 LET Y=YO
9920 LET D=1
9930 GOTO 9860

```


TRUCOS Y RUTINAS BASICAS



ONTINUANDO con programas y rutinas útiles al programador y que le permitan sacar más partido a la alta resolución de su ordenador, vamos a ver, en versión SPECTRUM, un programa para diseñar pantallas. Con

él dispondremos de todas las funciones del Basic del SPECTRUM, tales como dibujar puntos, líneas, círculos, etc.

La línea número 8, del siguiente programa, quedaría así:

```
8 DEF FN C$ (a, a$)=CHR$(22)+CAR$(a)+CHR$ ((32-LENa$)/2)+a$
```

```
1 REM *****
2 REM *
3 REM * DISENADOR DE PANTALLAS PARA SPECTRUM *
4 REM *
5 REM *****
6 REM
7 REM *** INICIALIZACION ***
8 REM
9 BORDER 0: PAPER 0: INK 6
10 CLEAR 48899
11 GO SUB 9800
12 CLS : RANDOMIZE USR 48900
13 LET X=128: LET Y=88
14 LET D=1
15 LET O=1
16 LET P=0: LET PL=0
17 LET TIN=7: LET PAP=0
18 GO TO 9500
19 REM
20 REM *****
22 REM * SITUA EL CURSOR EN SU NUEVA POSICION *
24 REM *****
25 REM
30 PLOT OVER 1: PAPER PAP:X,Y
40 LET X=X+(D AND I$="X" AND D+X<256)-(D AND I$="Z" AND X-D>-1)
50 LET Y=Y+(D AND I$="K" AND Y+D<176)-(D AND I$="M" AND Y-D>-1)
55 PLOT OVER 1: PAPER PAP:X,Y
60 INPUT "": PRINT #0:"X=";X;" ";TAB 7;"Y=";Y;" ";OVER:"";0
70 RETURN
80 REM
100 REM *****
101 REM * DIBUJO PROPIAMENTE DICHO *
102 REM *****
```

TRUCOS Y RUTINAS BASICAS

```

103 REM
110 LET I$=INKEY$: IF I$="" THEN GO TO 110
120 IF I$="Z" OR I$="X" OR I$="K" OR I$="M" THEN GO SUB 20: GO TO 110
130 IF I$="O" THEN LET O=(O=0): OVER O: BEEP .1,30: GO SUB 60: GO TO 110
140 IF I$="I" THEN BEEP .1,30: BEEP .1,32: BEEP .1,34: GO SUB 9000: GO TO 110
150 IF I$="P" THEN INPUT "PAPER ? ";T: IF T>-1 AND T<8 THEN LET PAP=T: BEEP .
1,30: PAPER PAP: GO TO 110
160 IF I$="T" THEN INPUT "TINTA ? ";T: IF T>-1 AND T<8 THEN LET TIN=T: BEEP .
1,30: INK T: GO TO 110
170 IF I$="G" THEN INPUT "GOTO X=";T1;" Y=";T2: IF FN H(T1) AND FN V(T2) THEN
PLOT OVER 1;X,Y: LET X=T1: LET Y=T2: PLOT OVER 1;X,Y: BEEP .1,30: GO SUB 60:
GO TO 110
180 IF I$="L" THEN PLOT X1,Y1: DRAW X-X1,Y-Y1: BEEP .1,30: LET X1=X: LET Y1=Y:
GO TO 110
190 IF I$="C" THEN INPUT "RADIO ? ";T: IF FN H(T+X) AND FN V(T+Y) THEN CIRCLE
X,Y,T: BEEP .1,30: GO TO 110
200 IF I$="V" THEN INPUT "VELOCIDAD DEL CURSOR ? ";T: IF FN V(T) THEN LET D=T
: BEEP .1,30: GO TO 110
210 IF I$="R" THEN RANDOMIZE USR 48900: BEEP .1,0: RETURN
220 IF I$="" THEN PLOT OVER 1;X,Y: LET X1=X: LET Y1=Y: FOR Q=1 TO 15: NEXT Q
: GO TO 110
800 GO TO 110
8000 REM
8010 REM *****
9000 REM * INSTRUCCIONES *
9010 REM *****
9011 REM
9014 RANDOMIZE USR 48900
9015 DEF FN C$(a,a$)=CHR$ 22+CHR$ a+CHR$ INT ((32-LEN a$)/2)+a$
9016 DEF FN H(T)=(T>-1 AND T<256): DEF FN V(T)=(T>-1 AND T<176)
9020 CLS : PRINT FN C$(0,"DISENADOR DE PANTALLAS"),FN C$(1,"-INSTRUCCIONES-")
9025 LET t=3
9030 PRINT : PRINT TAB t; INVERSE 1;"I"; INVERSE 0;"-Instrucciones."
9040 PRINT TAB t; INVERSE 1;"L"; INVERSE 0;"-Linea."
9050 PRINT TAB t; INVERSE 1;"C"; INVERSE 0;"-Circulo."
9060 PRINT TAB t; INVERSE 1;"P"; INVERSE 0;"-Cambia PAPER ."
9070 PRINT TAB t; INVERSE 1;"T"; INVERSE 0;"-Cambia INK ."
9080 PRINT TAB t; INVERSE 1;"V"; INVERSE 0;"-Cambia velocidad de cursor."
9090 PRINT TAB t; INVERSE 1;"G"; INVERSE 0;"-Goto punto."
9105 PRINT TAB T; INVERSE 1;"O"; INVERSE 0;"-Cambia OVER ."
9110 PRINT TAB t; INVERSE 1;"R"; INVERSE 0;"-Vuelve al menu principal."
9112 PRINT " "; INVERSE 1;"SPACE"; INVERSE 0;"-Dibuja punto."
9115 PRINT "TAB T;"Cursor:"; INVERSE 1;"X"; INVERSE 0;"-Derecha.", INVERSE 1;"
Z"; INVERSE 0;"-Izquierda.", INVERSE 1;"K"; INVERSE 0;"-Arriba.", INVERSE 1;"M
"; INVERSE 0;"-Abajo."
9120 PRINT #0;FN C$(0,"-PULSA UNA TECLA-")
9130 PAUSE 0
9131 INK TIN
9132 PAPER PAP
9133 OVER O
9140 RANDOMIZE USR 48912
9150 RETURN
9200 REM
9210 REM *****
9500 REM * MENU PRINCIPAL *
9510 REM *****
9511 REM
9520 POKE 23658,8
9521 INK 0
9522 PAPER 7

```

```

9523 CLS
9524 PRINT FN C$(0,"DISENADOR DE PANTALLAS")
9530 PRINT FN C$(2,"-ESCOGE OPCION-")
9540 PRINT : PRINT "'TAB 5;"1-DIBUJAR.'"TAB 5;"2-SAVE PANTALLA.'"TAB 5;"3-LOA
D PANTALLA.'"TAB 5;"5-BORRAR PANTALLA.'"TAB 5;"6-COPY DE PANTALLA.'"TAB 5;"7-
VOLVER AL BASIC."
9550 INPUT OPC
9560 IF OPC=1 THEN INK TIN: PAPER PAP: OVER O: RANDOMIZE USR 48912: PLOT OVER
1;X,Y: GO SUB 60: GO SUB 100: GO TO 9500
9570 IF OPC=2 THEN GO SUB 9630: GO TO 9520
9580 IF OPC=3 THEN GO SUB 9660: GO TO 9520
9590 IF OPC=7 THEN STOP
9600 IF OPC=5 THEN PAPER O: CLS : RANDOMIZE USR 48900: GO TO 9500
9610 IF OPC=6 THEN RANDOMIZE USR 48912: COPY
9620 GO TO 9500
9621 REM
9622 REM *****
9630 REM * SAVE PANTALLA *
9631 REM *****
9632 REM
9640 GO SUB 9700
9645 IF LEN N$>10 OR N$="" THEN RETURN
9650 RANDOMIZE USR 48912
9655 SAVE N$SCREEN$
9656 RETURN
9657 REM
9658 REM *****
9659 REM * LOAD PANTALLA *
9660 REM *****
9661 REM
9665 GO SUB 9700
9666 IF LEN N$>10 THEN RETURN
9670 LOAD N$SCREEN$
9671 INPUT ""
9672 RANDOMIZE USR 48900
9680 RETURN
9681 REM
9682 REM *****
9683 REM * ENTRADA DEL NOMBRE *
9684 REM *****
9685 REM
9700 INPUT "NOMBRE ? (MAX. 10 CARACTERES)'" LINE N$
9710 RETURN
9780 REM
9790 REM *****
9800 REM * RUTINAS C/M *
9805 REM *****
9810 REM
9820 FOR I=48900 TO 48923
9821 READ A
9822 POKE I,A
9823 NEXT I
9830 RETURN
9840 REM
9841 REM *****
9842 REM * LINEAS DATA *
9843 REM *****
9844 REM
9850 DATA 33,0.64,17,80,195,1,0,27,237,176,201,33,80,195,17,0.64,1,0,27,237,176,
201

```

TRUCOS Y RUTINAS BASICAS

Una vez introducido el programa, al ejecutarlo nos aparece una pantalla como la que se ve en la figura 1. El menú general.

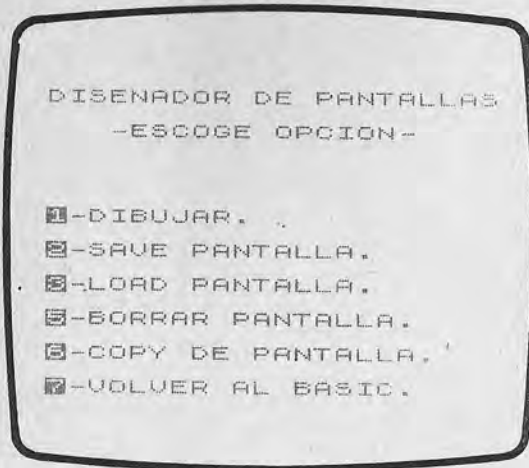


Fig. 1. Menú principal.

Las opciones de que disponemos son las siguientes:

1. DIBUJAR. Pulsando esta opción entramos directamente en la pantalla de diseño. Dentro de esta pantalla dispondremos de una serie de comandos para poder dibujar.

2. SAVE PANTALLA. Grabamos la pantalla que acabamos de realizar en cinta de casete.

3. LOAD PANTALLA. Cargamos una pantalla que se encuentra almacenada en una cinta de casete.

4. BORRAR PANTALLA. Borrarnos la pantalla de diseño y restauramos los valores del papel. El papel se volverá a poner, en caso de que hubiese cambiado, de color negro (PAPER O).

5. COPY DE PANTALLA. Se realiza un copy de la pantalla en la impresora. En el caso de que tengamos un interface RS232 de Indescomp, habrá que cambiar la línea 9610 por:

```

9610 IF OPC=6 THEN RANDOMIZE USR
48912: RANDOMIZE USR 65044
    
```

6. VOLVER AL BASIC. Terminamos el programa.

Entramos en la parte más importante del programa pulsando la opción 1 del menú principal. Al hacerlo la pantalla se borrará, y la veremos totalmente vacía a excepción de la última línea, que nos indica las coordenadas de cursor gráfico y del estado del over. También veremos un puntito muy pequeño situado, aproximadamente, en el centro de la pantalla. Este pequeño punto es lo que llamaremos el

cursor gráfico y servirá para que el usuario sepa dónde está dibujando.

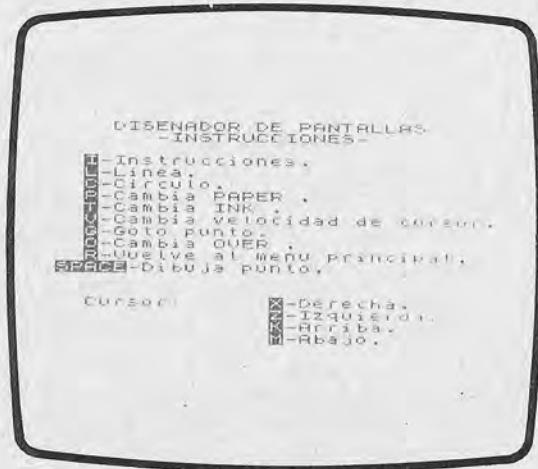


Fig. 2. Resumen de comandos que se pueden utilizar.

Para mover el cursor utilizaremos las siguientes tablas:

X: Derecha.
Z: Izquierda.
K: Arriba.
M: Abajo.

En la figura 2 podemos ver un resumen de los comandos que podemos utilizar. Para poder visualizar esta pantalla en cualquier momento de nuestro diseño, sólo tenemos que pulsar la tecla "T" para verla. El contenido de nuestra pantalla, aunque no lo parezca, se almacena y no se pierde. Por eso, una vez vistos los comandos, al pulsar una tecla, volvemos a la pantalla de diseño y volveremos a ver nuestro dibujo.

Vamos a ver ahora una pequeña explicación de los comandos disponibles.

I. Muestra la pantalla que se ve en la figura 2.

L. Traza una línea desde el último punto o desde la última línea trazada.

C. Nos dibuja un círculo. El ordenador nos pedirá el radio que queremos que tenga.

P. Cambia el color del papel. Al cambiarlo, según vamos moviendo el cursor, el papel cambiará de color.

T. Cambia el color de la tinta. A partir del momento en que cambiemos el color de la tinta, todo lo que dibujemos aparecerá del nuevo color.

G. Nos permite colocar el cursor gráfico en cualquiera de la pantalla sin dibujar. Esto nos permite no tener que ir moviendo el

cursor de una punta a otra de la pantalla con las teclas de movimiento.

O. Nos cambia la cualidad del OVER. Si estamos en OVER 1, pasaremos a OVER 0 y si estuviésemos en OVER 0, pasaremos a OVER 1.

R. Nos permite volver al menú principal. Aunque se borre la pantalla, no hay ningún problema. No ha desaparecido.

SPACE. Dibujamos, o borramos, el punto sobre el que se encuentra el cursor gráfico. Si estamos en OVER 1, al pulsar la tecla SPACE sobre un punto iluminado lo borraremos.

Cabe la posibilidad de que el propio usuario incluya alguna que otra función nueva en el programa. Por ejemplo, si queremos poder utilizar el comando BRIGHT para, así, tener más colores, podríamos incluir las siguientes líneas:

```
15 LET O=1:LET B=0
125 IF I$="B" THEN B=(B=B):BRIGHT
    B:BEEP .1,30:GOSUB 60
```

Si lo que queremos es poder utilizar la función FLASH, para poder hacer dibujos que parpadeen, tendremos que incluir las siguientes líneas:

```
14 LET O=1:LET F=0
127 IF I$="F" THEN F=(F=F):FLASH
    F:BEEP .1,30:GOSUB 60
```

Cuantas más funciones incluyamos en el programa, más completo será. Por otro lado, si queremos tener información continua sobre el estado del OVER, del BRIGHT y del FLASH, podemos variar la línea 60 como sigue:

```
60 INPUT "":PRINT #0;"X=";X;" ";TAB
    7;"Y=";Y;" ";OV:"O," BR:"B," FL:"F
```

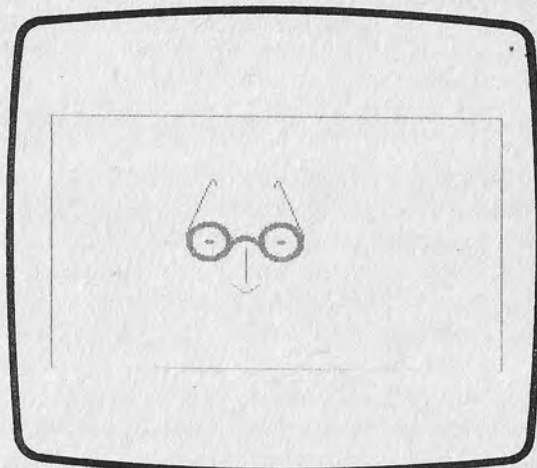
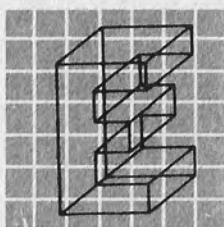


Fig. 3. Un pequeño dibujo de administración.

■ Tarjeta digitalizadora de vídeo

INTRODUCCION



N el presente artículo presentaremos una tarjeta que con el software que se describirá en el próximo, permitirá digitalizar una señal de vídeo compuesto y visualizar el resultado obtenido en el monitor de un ordenador personal. El tiempo requerido para todo el proceso será alrededor de medio segundo. Como generador de señal de vídeo se podrá utilizar una cámara de vídeo, un vídeo-casete recorder, o incluso la señal de vídeo procedente de otro ordenador. En cuanto al ordenador a utilizar deberá poseer un controlador de CRT que disponga de la posibilidad de gráficos de alta resolución (320×200 mínimo) en blanco y negro, circunstancia que es habitual en la mayoría de los ordenadores personales existentes. Pero previamente a presentar y comentar el diseño, daremos una visión general de la problemática asociada al proceso que

deberá seguir la señal de vídeo para su almacenamiento en el ordenador.

■ Características de la señal de vídeo

La forma específica de una señal de vídeo compuesto está íntimamente ligada con el modo de formación de la imagen en un monitor o pantalla CRT, por lo que a continuación se dará una sencilla descripción del funcionamiento de los tubos CRT (tubo de rayos catódicos).

En un visualizador CRT (figura 1), un filamento incandescente emite un haz de electrones.

Al incidir dicho haz contra la capa fluorescente existente en la zona frontal rectangular (pantalla), se produce un punto luminoso, cuya intensidad se puede controlar variando el flujo de electrones, permitiendo con ello tener una determinada gama de grises. La posición del punto luminoso en la pantalla se puede modificar mediante deflexiones horizontales y verticales del mismo, originados por campos electromagnéticos.

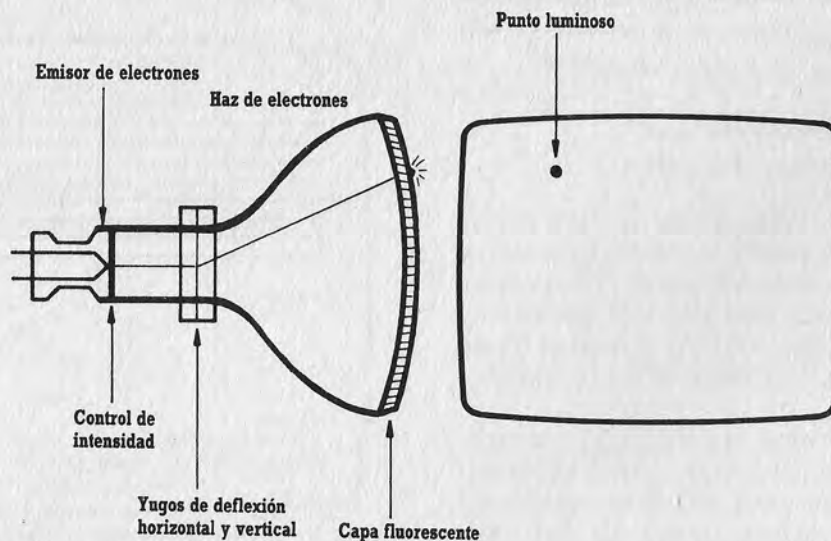


Fig. 1. Esquema de pantalla de CRT.

EL TALLER DEL HARDWARE

La imagen en el monitor se forma al recorrer el haz de electrones la pantalla horizontalmente (figura 2) desde izquierda a derecha. Al llegar al extremo derecho se disminuye la intensidad del haz, para que no sea perceptible, y se retorna al extremo opuesto bajando una línea en la pantalla. Al trazar la última línea horizontal se hace retornar el pincel luminoso al extremo superior izquierdo con el fin de repetir el proceso anterior. La frecuencia del haz horizontal es de 15.625 Hz y la del vertical 50 Hz, de donde el número de líneas que describe el haz son 312,5 por cada pantalla, cuadro o trama. Estas frecuencias corresponden a las normas europeas PAL y SECAM. Para los equipos americanos son de aplicación las normas NTSC, que difieren en varios aspectos de las europeas:

- Período de barrido de línea: 63,5 microsegundos, frente a 64.
- Período de barrido de cuadro: 16,6 milisegundos, frente a 20.
- Codificación de la señal color diferente.

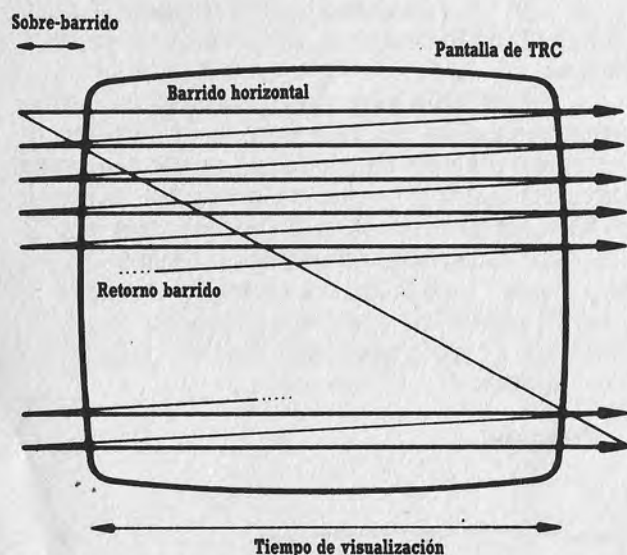
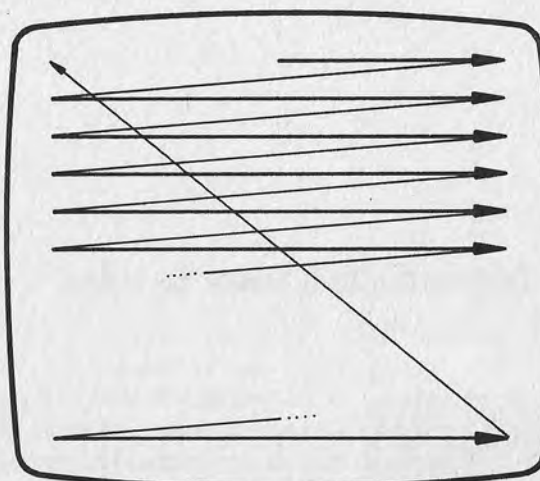


Fig. 2. Formación de la imagen en la pantalla.

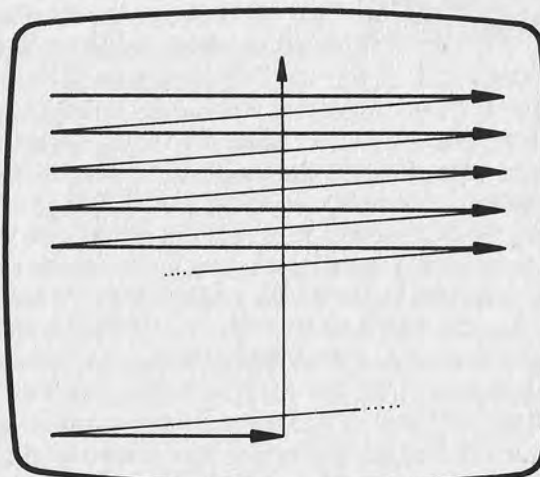
El tipo de barrido de la pantalla por el pincel electrónico descrito se conoce como simple o directo; sin embargo, en los receptores de TV comerciales se utiliza el denominado barrido entrelazado (figura 3), con el fin de conseguir un mayor número de líneas verticales.

Como se observa en la figura, dos campos de 312,5 líneas se utilizan para describir una imagen completa con 625 líneas (cuadro). Cada línea de un campo queda en una posición intermedia o «entrelazada» con dos líneas

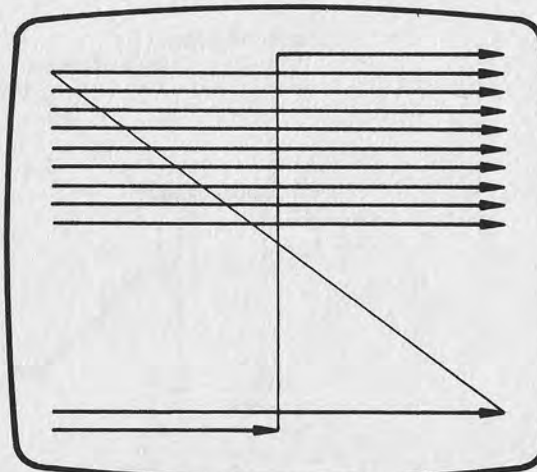
del campo anterior, lo que da el nombre a este tipo de barrido.



Campo 1 (312,5 líneas)



Campo 2 (312,5 líneas)



Campos combinados = Cuadro 625 líneas

Fig. 3. Barrido entrelazado.

Existen otros equipos que utilizan activación directa de cada cañón luminoso denominándose consecuentemente de ataque directo (*Direct drive*). Suelen llevar por separado las señales de sincronismo o superponerlo a uno de los canales de color.

Con la descripción anterior, estamos en condiciones de comprender la particular forma de una señal de vídeo compuesto (fig. 4). Dicha señal consiste en una secuencia de líneas de información. Cada línea se corresponde con un trazado horizontal del pincel electrónico en el receptor CRT. De los 64 microsegundos de duración de estas líneas, solamente unos 52 microsegundos son visibles, durante los cuales los diferentes niveles de tensión de la señal controlan la intensidad del haz de electrones (figura 1), haciendo aparecer sobre la línea puntos luminosos de mayor o menor intensidad según el nivel de tensión existente. Los restantes 12 microsegundos se utilizan en el sobrebarrido (pórticos frontal y posterior) y retorno del haz electrónico en el monitor (impulso de sincronismo horizontal) (figura 2).

Como se mencionó anteriormente, un grupo de 312,5 líneas corresponden a un campo. Cada campo está separado del siguiente y anterior por cinco impulsos de sincronismo vertical cuya duración es de 26,8 microsegundos. Como se muestra en la figura 4, los dos campos quedan diferenciados porque a uno de ellos se le añade 1/2 línea en la zona correspondiente a los impulsos de sincronis-

mo horizontal. De este modo el terminal receptor puede entrelazar las informaciones de las líneas de vídeo correspondientes a cada uno de los dos campos consecutivos secuencialmente recibidos en orden a constituir un cuadro de 625 líneas.

De lo anteriormente expuesto se comprende la denominación «vídeo compuesto» de la señal descrita, ya que se compone de información moduladora del haz electrónico (señal vídeo) y de sincronismos horizontales y verticales para conseguir el retorno del haz hacia la izquierda y hacia la parte superior de la pantalla, respectivamente.

Los monitores de altas prestaciones para aplicaciones de diseño asistido por ordenador suelen emplear frecuencias de barrido más altas y sin entrelazado, con objeto de dar mayor estabilidad a la imagen y, por tanto, mayor confort al que las emplea.

Proceso de la señal de vídeo

Una vez conocidas las propiedades y forma de la señal de vídeo habrá que adecuar y trasladar la información contenida en la misma al ordenador. La señal de vídeo es una señal analógica y, como sabemos, un ordenador sólo admite información digital, lo que da lugar a una serie de cuestiones cuyas respuestas se presentarán en los siguientes párrafos.

El primer problema a resolver es deci-

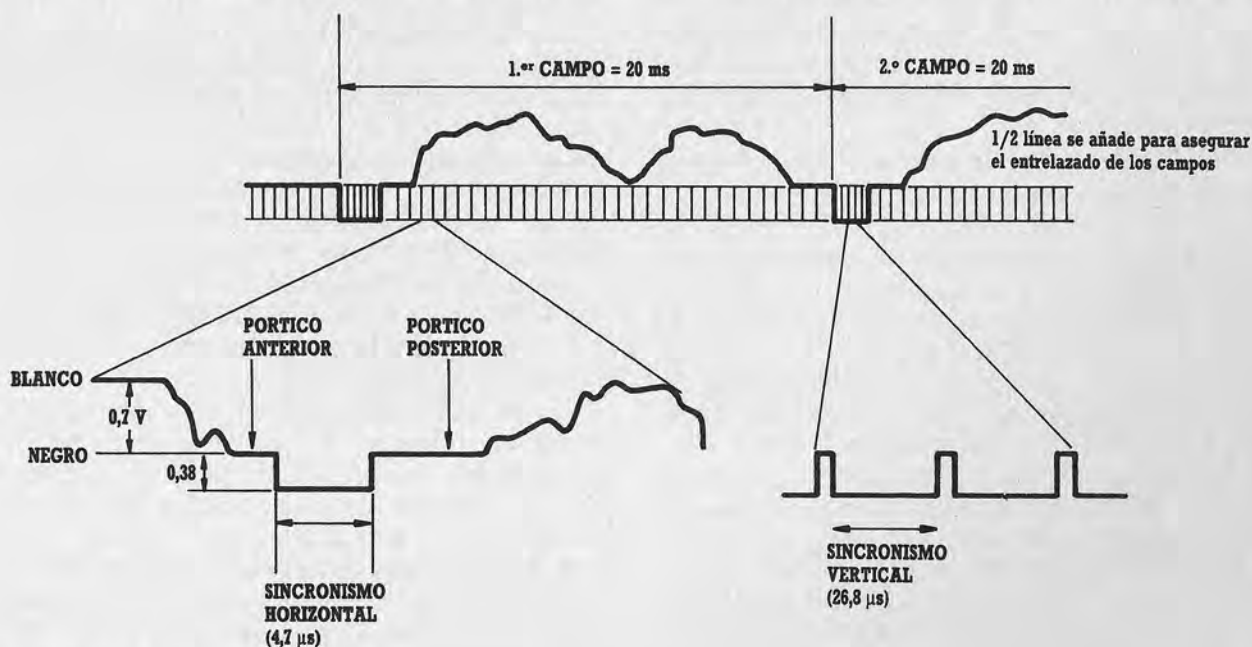


Fig. 4. Señal de vídeo y detalles.

dir la resolución espacial que dispondrá la imagen digital resultante, es decir, cuántas muestras de la señal de vídeo se tomarán. De las 312,5 líneas que componen un campo, es práctica normal almacenar 256 líneas, ya que es la potencia de dos más próxima y ésta será la resolución vertical adoptada en el diseño. En cuanto al número de muestras por línea se tomarán también 256, con lo que la imagen tendrá una resolución de 256×256 . Al constituir cada una de estas muestras un elemento de la imagen digital que se obtendrá en el ordenador se les denomina «pixels», que es una contracción del término inglés «picture element».

La segunda cuestión a abordar es que, como sabemos, la señal de vídeo es una señal analógica, es decir, que contiene un número infinito de valores posibles de la tensión, que representan otros tantos de intensidades luminosas, como ya se comentó, lo que implica que habrá que cuantificarla para su almacenamiento. Para ello los pixels anteriormente muestreados se digitalizan mediante un convertidor analógico-digital, que se encarga de transformar el valor analógico de los pixels en números binarios equivalentes y con formato apropiado para su proceso en un computador digital.

Con el fin de simplificar el tratamiento analógico de la señal de vídeo en nuestra aplicación, se utilizará un C A/D de 1 bit de resolución. Un C A/D de un bit es un comparador que se puede realizar mediante un amplificador operacional. Las imágenes digitales que se obtienen con este tipo de C A/D se denominan binarias.

Un aspecto importante a analizar es el tiempo que se tomará para la adquisición de las 256×256 muestras. Como los 256 pixels de cada línea se han de muestrear en el período activo de la línea de vídeo, esto implica que si éstos se tomaran consecutivamente en los 52 microsegundos disponibles, el tiempo de adquisición para cada pixel sería de $52 \text{ microsegundos} / 256 = 203 \text{ ns}$. Para poder soportar esta alta velocidad de transferencia de datos, la interfaz debería o bien contener una memoria de alta velocidad o bien realizarla por acceso directo a memoria, pero con el fin de simplificar el diseño, se adoptará la solución de que sea el propio microprocesador del ordenador utilizado el encargado de realizar dicha transferencia desde la interfaz a la memoria del ordenador personal. Esto implicará que los 256

pixels de cada línea se adquirirán en varios campos, lo que únicamente traerá como consecuencia que la imagen física a digitalizar no deberá cambiar durante el tiempo que dure su digitalización con el fin de obtener un buen resultado. Como se verá posteriormente, la mayoría de los microprocesadores, con la estructura de interfaz que se describirá digitalizarán la imagen en 32 cuadros, que equivalen a 0,64 segundos.

■ Diagrama de bloques de la interfaz

Con la información contenida en los párrafos anteriores, estamos en condiciones de plantear el diagrama de bloques de la interfaz que se presenta. Como se observa en la figura 5, la señal de vídeo compuesto llega a un separador que descompone dicha señal en sus tres componentes básicas: la de vídeo propiamente dicha, y las señales de sincronismo vertical y horizontal, que nos sirven para la puesta a cero de los circuitos contadores denominados X e Y.

El contador Y se resetea con los pulsos de sincronismo vertical (50 Hz) y se incrementa con los pulsos de sincronismo horizontal (15.750 Hz), y de este modo contiene la posición vertical de la línea de barrido con respecto al origen del cuadro; por tanto, su valor en cada instante es el número de línea o coordenada Y.

El contador X es un contador que se resetea con el impulso de sincronismo horizontal y se incrementa por el reloj de punto CKP de alta frecuencia (5 MHz). Este contador contiene, por tanto, la posición que tiene en cada instante dentro de una línea horizontal determinada el haz electrónico que explora la pantalla. Esta será nuestra coordenada X. Estos contadores nos permiten considerar el monitor donde se visualiza la imagen como el sistema de coordenadas que se representa en la figura 6.

Si se desea saber el valor digital de la señal de vídeo en un conjunto de puntos de coordenadas desde (X-Y) hasta (X+7, Y), se deberá escribir las coordenadas del primero de ellos (X,Y) en los puertos de salida X e Y, respectivamente, y cuando los contadores de posición de barrido sean iguales a la dirección elegida, se genera un pulso (EQU), que hace que el sistema, mediante un contador, cuente 8 pulsos de reloj de punto y a la aparición del

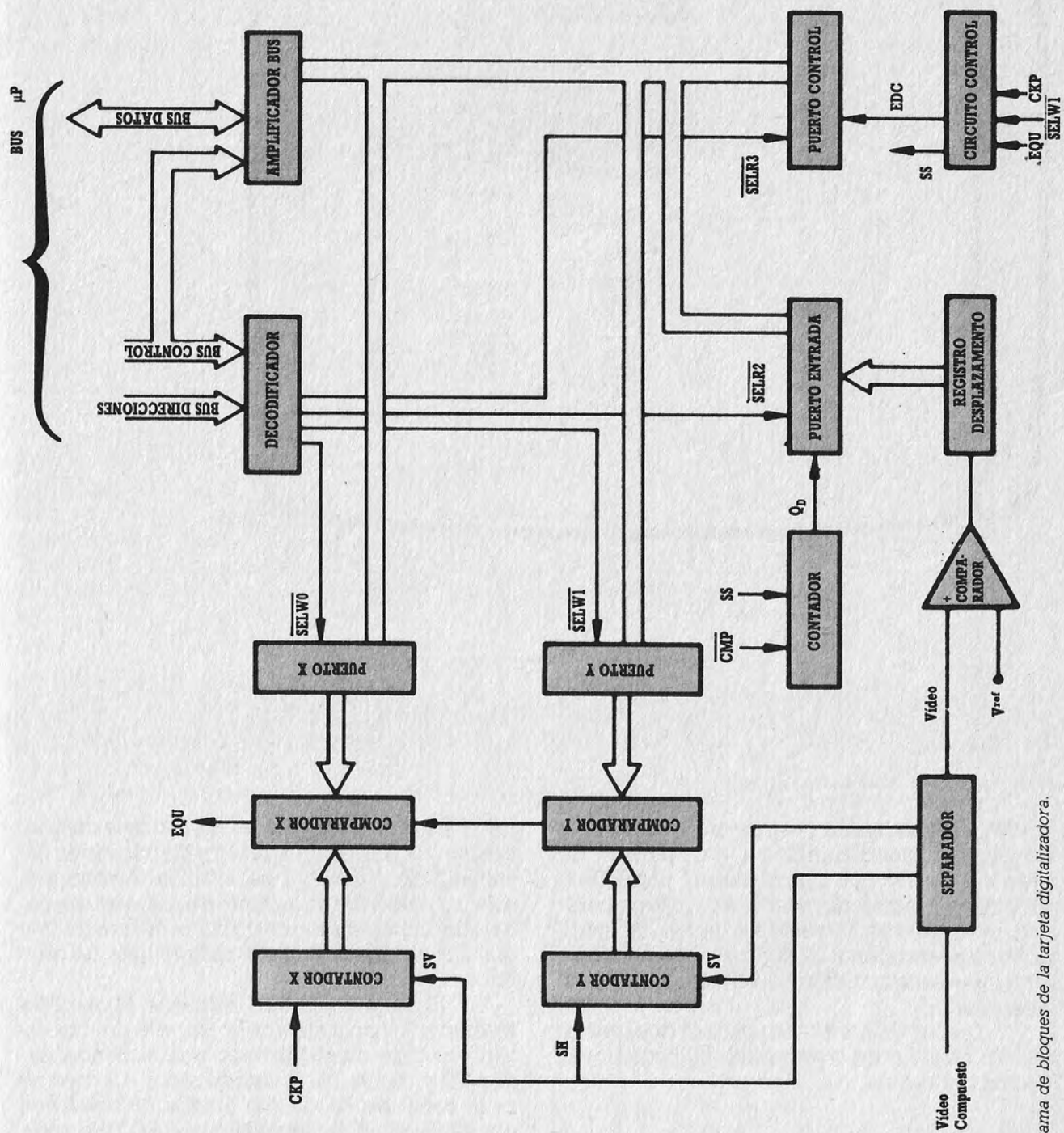


Fig. 5. Diagrama de bloques de la tarjeta digitalizadora.

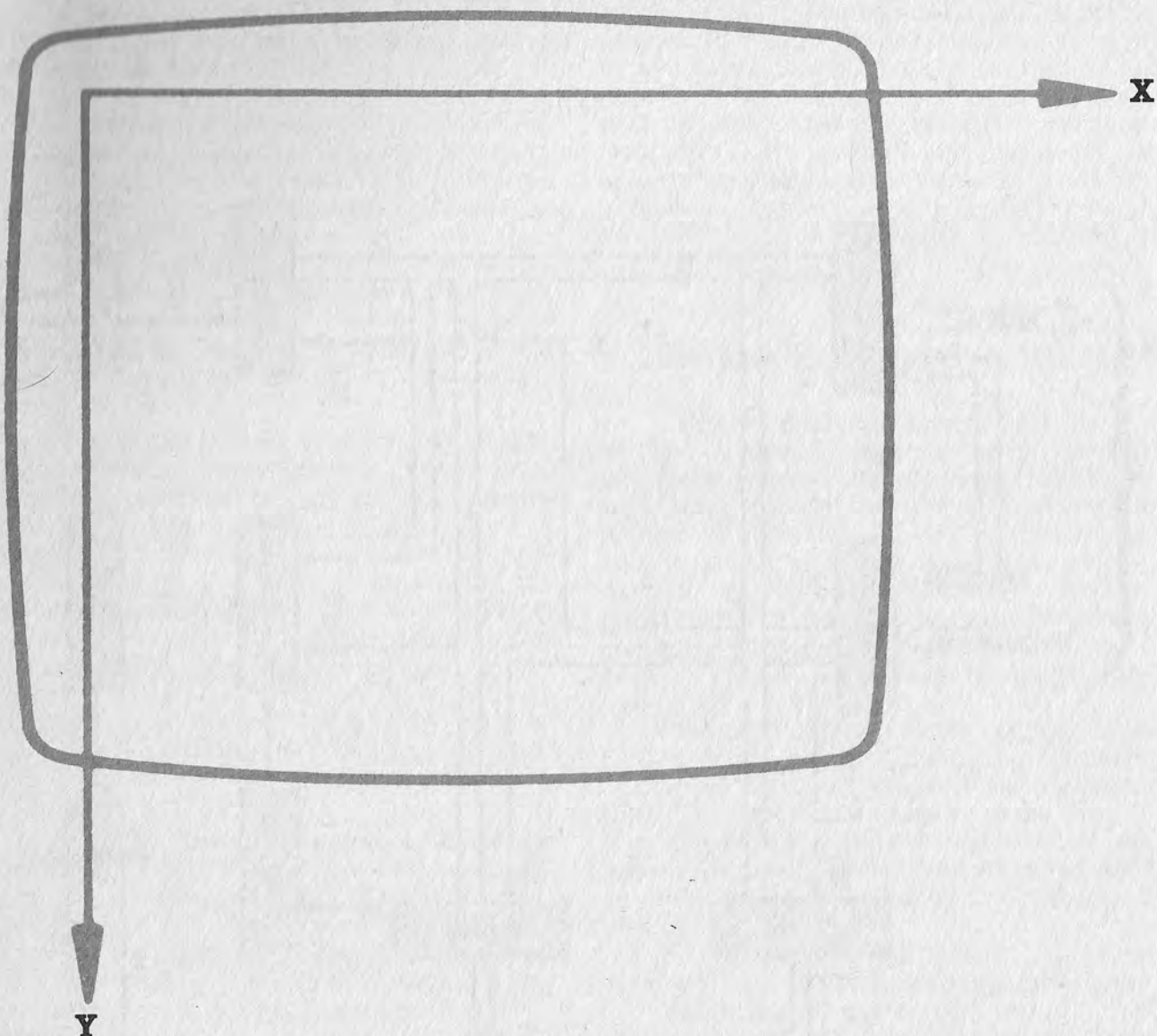


Fig. 6. Sistema de coordenadas empleado.

oétavo se almacene en el puerto de entrada la información procedente de un registro de desplazamiento que convierte en paralelo la información serie de salida de un comparador, que binariza la señal de vídeo. A continuación se pormenorizará cada uno de los bloques que someramente se han comentado en este apartado.

Distinguiremos dos partes diferenciadas: el circuito de tratamiento analógico y el de control digital.

■ Circuito de tratamiento analógico

Este circuito que se representa en la figura 7 tiene como misiones descomponer la

señal de vídeo compuesto en sus tres componentes, así como binarizar la señal de vídeo resultante de la anterior separación. A continuación se describe la misión de los componentes fundamentales del circuito. El condensador C1 elimina la posible componente de continua de la señal.

El amplificador A1 aumenta el nivel de tensión a 2 Vpp y separa la entrada del circuito fijador de nivel formado por el condensador C2 y diodo D1. El comparador A3 compara la señal de salida del fijador de nivel con una referencia de tensión variable. Esta referencia de tensión se tendrá que ajustar mediante el potenciómetro P1, de manera que dicho comparador esté saturado positivamente únicamente durante los impulsos de sincronis-

EL TALLER DEL HARDWARE

mo. El condensador C3 alarga los pulsos generados por el comparador A2, de manera que los pulsos de sincronismo horizontal supe-

ren el pedestal posterior. Asimismo su presencia conforma los cinco pulsos de sincronismo vertical en uno solo.



Fig. 8. Tren de pulsos a la salida de U18.

Las dos puertas inversoras en cascada eliminan los posibles ruidos generados en el proceso anterior. A la salida de dichas puertas se obtiene el tren de pulsos de la figura 8, en el que están mezclados los pulsos de sincronismo vertical y horizontal.

Los dos conjuntos biestable-monoestable (U20 y U21, respectivamente, en figura 7) separan el anterior tren de pulsos por la duración de los mismos.

El monoestable M2 tiene una duración en su estado cuasiestable de 10 microsegundos, de manera que discrimina los pulsos de sincronismo horizontal, ya que son de menor duración, conteniendo la salida de su biestable B2 asociando únicamente pulso de sincronismo vertical. La duración del estado cuasiestable de M1 es 1 microsegundo, con lo que elimina cualquier pulso esporádico de menor duración que aparezca en la entrada. La salida de su biestable asociado B1 contiene pulsos de sincronismo horizontal y vertical, aunque esto no afecta al diseño.

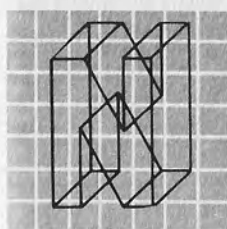
Por último, el comparador A2 binariza la señal de vídeo al comparar dicha señal con la tensión de referencia fijada por el potenciómetro P2. Cuando la señal de vídeo sea superior a la tensión de referencia, dicho comparador estará saturado positivamente (1), estando en el estado de saturación negativa en el caso contrario (0). Ajustando el potenciómetro P2, conseguirá obtener una imagen con mayor o menor número de pixels blancos en el monitor del computador, es decir, que hace una función similar al mando de brillo en los visualizadores CRT.

■ Circuito de control digital

Los circuitos necesarios para controlar la digitalización serán descritos en el próximo tomo, junto con un programa de adquisición y almacenamiento de los datos.

NATURALEZA Y TECNOLOGIA

■ Cantidad de calor



OSOTROS somos capaces de percibir la temperatura de los cuerpos a través del tacto; y decimos que un cuerpo está más o menos caliente, pero no es el calor lo que realmente medimos, sino la temperatura de ese

cuerpo.

El calor es la energía en movimiento entre dos cuerpos que se encuentran a distinta temperatura; por tanto, sólo podemos calcular el calor como un diferencial, o si no como la cantidad de calor que gana o pierde un cuerpo que sufre un incremento de temperatura, ya sea positivo o negativo.

En el programa que a continuación presentamos se estudia la cantidad de calor que gana o pierde un cuerpo cuando sufre un incremento de temperatura. Para ello utilizamos las relaciones que nos da la siguiente fórmula:

$$Q = m \cdot c \cdot (t - t_0)$$

Y que son:

- Las cantidades de calor son directamente proporcionales a los incrementos de temperatura.

- También son directamente proporcionales a las masas.

- Y dependen también de la sustancia de la que se trate.

El programa edita primeramente un menú en el que habrá que señalar en qué es-

tado se encuentra el cuerpo del que estemos tratando.

Según la sustancia, deberá ponerse el calor específico correspondiente. El programa muestra algunos de los calores más corrientes, pero si el del cuerpo elegido no se encuentra entre ellos, deberá ser introducido, pulsando la opción 6-OTROS.

Esta fórmula que utilizamos puede ser comprobada experimentalmente con un aparato llamado calorímetro. Consta de dos recipientes (uno menor metido dentro de otro mayor) aislados entre sí con el fin de que el calor no se propague entre ellos y no se pierda; ambos están tapados herméticamente y en la tapadera tiene dos orificios por los cuales se introducen un removedor y un termómetro para poder controlar la temperatura del cuerpo que se introduce en el interior.

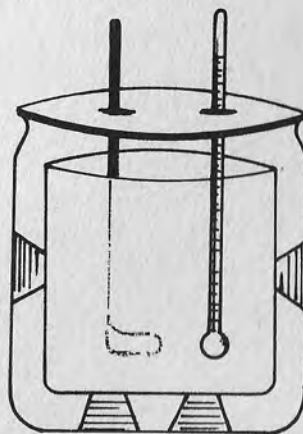


Fig. 1. Calorímetro.

APRENDER CON EL ORDENADOR

```

10 REM *****
20 REM ** CANTIDADES DE CALOR **
30 REM *****
40 DIM G$(6):DIM L$(6):DIM S$(9):DIM G(6):DIM L(6):DIM S(9)
50 REM *****
60 REM ** LECTURA DE DATOS **
70 REM *****
80 FOR I=1 TO 6
90 READ G$(I)
100 NEXT I
110 FOR I=1 TO 6
120 READ L$(I)
130 NEXT I
140 FOR I=1 TO 9
150 READ S$(I)
160 NEXT I
170 FOR I=1 TO 6
180 READ G(I)
190 NEXT I
200 FOR I=1 TO 6
210 READ L(I)
220 NEXT I
230 FOR I=1 TO 9
240 READ S(I)
250 NEXT I
400 REM *****
410 REM ** PROGRAMA PRINCIPAL **
420 REM *****
430 CLS
440 PRINT "***** CANTIDADES DE CALOR *****"
450 PRINT:PRINT
460 PRINT "TU MATERIAL ES:"
470 PRINT:PRINT:PRINT
480 PRINT "1.-GAS"
490 PRINT:PRINT
500 PRINT "2.-LIQUIDO"
510 PRINT:PRINT
520 PRINT "3.-SOLIDO"
530 PRINT:PRINT
540 INPUT "INTRODUCE EL NUMERO CORRESPONDIENTE A TU MATERIAL";A
550 IF A<1 OR A>3 THEN GOTO 430
560 CLS
570 IF A=1 THEN GOTO 830
580 IF A=2 THEN GOTO 1130
590 GOTO 1430
800 REM *****
810 REM ** GASES **
820 REM *****
830 CLS
840 PRINT "*** G A S E S ***"
850 PRINT:PRINT
860 FOR I=1 TO 6
870 PRINT " ";I;"-";G$(I):PRINT
880 NEXT I
890 INPUT "PULSA OPCION";O
900 IF O<1 OR O>6 THEN GOTO 830
910 PRINT:PRINT
920 IF O=6 THEN INPUT "INTRODUCE EL NOMBRE DE TU MATERIAL";G$(6):PRINT "INTRODUC
E EL CALOR ESPECIFICO EN cal/g.grado centigrado DE ";G$(6):INPUT G(6)
930 CLS
940 PRINT "MASA DE ";G$(O);" (en gramos) ":INPUT M
950 PRINT
960 PRINT "TEMPERATURA INICIAL DE ";G$(O);" (en grados centigrados) ":INPUT T0
970 PRINT
980 PRINT "TEMPERATURA FINAL DE ";G$(O);" (en grados centigrados) ":INPUT T1
990 LET Q=G(O)*M*(T0-T1)
1000 GOSUB 2000
1010 GOTO 3000
1100 REM *****
1110 REM ** LIQUIDOS **
1120 REM *****
1130 CLS
1140 PRINT "*** L I Q U I D O S ***"
1150 PRINT:PRINT
1160 FOR I=1 TO 6
1170 PRINT " ";I;"-";L$(I):PRINT
1180 NEXT I
1190 INPUT "PULSA OPCION";O
1200 IF O<1 OR O>6 THEN GOTO 1130
1210 PRINT:PRINT

```

```

1220 IF Q=6 THEN INPUT "INTRODUCE EL NOMBRE DE TU MATERIAL";L$(6):PRINT "INTRODU
DE EL CALOR ESPECIFICO EN cal/g.grado centigrado DE ";L$(6):INPUT L$(6)
1230 CLS
1240 PRINT "MASA DE ";L$(0); " (en gramos) ":INPUT M
1250 PRINT
1260 PRINT "TEMPERATURA INICIAL DE ";L$(0); " (en grados centigrados) ":INPUT T0
1270 PRINT
1280 PRINT "TEMPERATURA FINAL DE ";L$(0); " (en grados centigrados) ":INPUT T1
1290 LET Q=L$(0)*M*(T0-T1)
1300 GOSUB 2000
1310 GOTO 3000
1400 REM *****
1410 REM ** SOLIDOS **
1420 REM *****
1430 CLS
1440 PRINT "          *** S O L I D O S ***"
1450 PRINT:PRINT
1460 FOR I=1 TO 9
1470 PRINT "          ";I;".-";S$(I):PRINT
1480 NEXT I
1490 INPUT "PULSA OPCION";O
1500 IF O<1 OR O>9 THEN GOTO 1430
1510 PRINT:PRINT
1520 IF O=9 THEN INPUT "INTRODUCE EL NOMBRE DE TU MATERIAL";S$(0):PRINT "INTRODU
DE EL CALOR ESPECIFICO EN cal/g.grado centigrado DE ";S$(0):INPUT S$(0)
1530 CLS
1540 PRINT "MASA DE ";S$(0); " (en gramos) ":INPUT M
1550 PRINT
1560 PRINT "TEMPERATURA INICIAL DE ";S$(0); " (en grados centigrados) ":INPUT T0
1570 PRINT
1580 PRINT "TEMPERATURA FINAL DE ";S$(0); " (en grados centigrados) ":INPUT T1
1590 LET Q=S$(0)*M*(T0-T1)
1600 GOSUB 2000
1610 GOTO 3000
2000 REM *****
2010 REM ** CALOR RECIBIDO O PERDIDO **
2020 REM *****
2030 IF Q<0 THEN LET X$="RECIBIDO":GOTO 2050
2040 LET X$="PERDIDO"
2050 PRINT:PRINT
2060 PRINT "          ** CALOR ";X$;" =";ABS(Q); " calorías **"
2070 RETURN
3000 REM *****
3010 REM ** FIN DEL PROGRAMA **
3020 REM *****
3030 PRINT:PRINT:PRINT
3040 PRINT "          ** PULSA UNA TECLA PARA CONTINUAR **"
3050 PRINT "          ( S PARA SALIR )"
3060 LET Z$=INKEY$:IF Z$="" THEN GOTO 3060
3070 IF Z$="S" OR Z$="s" THEN END
3080 CLEAR:GOTO 10
6000 REM *****
6010 REM ** DATOS DE CALORES ESPECIFICOS **
6020 REM *****
6030 DATA "HIDROGENO","HELIO","AIRES","ANHIDRIDO CARBONICO","VAPOR DE AGUA","OTRO
S"
6040 DATA "AGUA","ALCOHOL","GLICERINA","BENCENO","TETRACLORURO DE BENCENO","OTRO
S"
6050 DATA "HIELO","SILICATOS","ALUMINIO","HIERRO","ESTANO","COBRE","MERCURIO","P
LOMO","OTROS"
6060 DATA 3.4,1.25,0.24,0.2,0.46,0.1,0.59,0.58,0.45,0.2,0.5,0.19,0.21,0.11,0.0
5,0.09,0.03,0.03,0

```

Modificaciones para otros equipos

El programa es válido para los ordenadores AMSTRAD, MSX, IBM PC y compatibles.

Las modificaciones para el resto de equipos son las siguientes:

SPECTRUM

```

40 DIM G$(6,19):DIM L$(6,23):DIM
S$(9,9):DIM G(6):DIM L(6):DIM S(9).

```

```

3070 IF Z$="S" OR Z$="s" THEN GOTO
9999.

```

COMMODORE

Cambiar CLS por PRINT CHR\$(147) en líneas 430, 560, 830, 930, 1130, 1230, 1430, 1530.

MATEMATICAS

Distancias en el espacio euclídeo (I)

Desde el tiempo de los griegos ya el hombre se interesaba por las distancias, in-

APRENDER CON EL ORDENADOR

ventaron muchos sistemas métricos para poder calcularlas. Pero no era suficiente este sistema experimental, era necesario un sistema que sirviese para todos los casos y que no dependiera de otra cosa que de la situación en el espacio de los puntos.

Para crear este sistema que opere en el espacio euclídeo es necesario únicamente un sistema de referencia en el espacio, unos vectores y un origen sobre el que basar todos los puntos.

Este sistema geométrico quedó perfectamente confeccionado con operaciones tan importantes como: el producto escalar entre vectores, el producto vectorial y el producto mixto.

En el programa que a continuación presentamos se resuelven muchos de estos problemas como son:

- Distancia entre dos puntos.

- Distancia entre un punto y una recta.
- Distancia entre dos rectas.

El programa pide en todos los casos las coordenadas de los puntos, que deberán ser introducidas sucesivamente, según el orden X, Y, Z.

En el caso de las rectas, se piden dos puntos pertenecientes a ellas, con las que se hallará el vector dirección de cada una.

Para hallar la distancia entre un punto y una recta utilizamos el producto vectorial con el objeto de calcular el área del triángulo formado por la recta y los vectores que unen el punto con la recta.

Por último, se hace un pequeño croquis en pantalla de la situación, presentando asimismo el resultado de la distancia.

En las figuras siguientes se pueden observar gráficamente las opciones del programa.

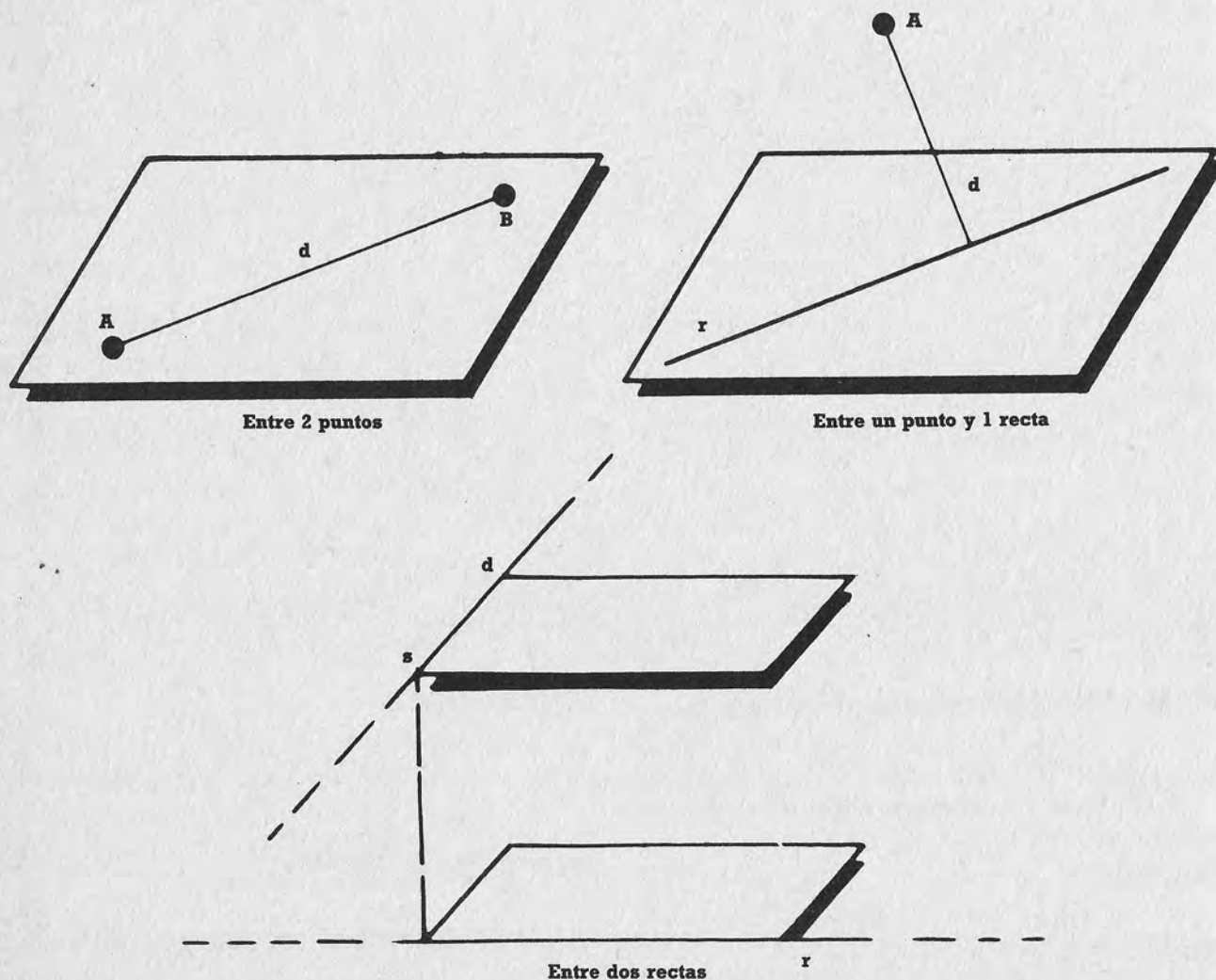


Fig. 2. Distancias.

```

10 SCREEN 1
20 REM *****
30 REM ** M E N U   P R I N C I P A L **
40 REM *****
50 CLS
60 LOCATE 2,16:PRINT "-----"
70 LOCATE 3,16:PRINT "          DISTANCIAS EN EL PLANO EUCLIDEO          "
80 LOCATE 4,16:PRINT "-----"
90 LOCATE 8,22:PRINT "A) Distancia entre dos puntos"
100 LOCATE 10,22:PRINT "B) Distancia entre un punto y una recta"
110 LOCATE 12,22:PRINT "C) Distancia entre dos rectas"
120 LOCATE 18,23:PRINT "*** PULSA LA OPCION ELEGIDA ***"
130 LET O$=INKEY$:IF O$="" THEN GOTO 130
140 IF O$<>"A" AND O$<>"B" AND O$<>"C" THEN GOTO 130
150 CLS
160 IF O$="A" THEN GOTO 190
170 IF O$="B" THEN GOTO 340
180 GOTO 540
190 REM *****
200 REM ** DISTANCIA ENTRE DOS PUNTOS **
210 REM *****
220 GOSUB 950
230 LET A=1:GOSUB 750
240 CLS:GOSUB 950:LET A=2:GOSUB 750
250 CLS:GOSUB 950
260 LOCATE 16,10:PRINT "(";X(1);",";Y(1);",";Z(1);")"
270 LOCATE 17,50:PRINT "(";X(2);",";Y(2);",";Z(2);")"
280 LET X=(X(1)-X(2))^2
290 LET Y=(Y(1)-Y(2))^2
300 LET Z=(Z(1)-Z(2))^2
310 LOCATE 12,38:PRINT SQR(X+Y+Z)
320 LINE (150,200)-(420,90)
330 GOTO 1070
340 REM *****
350 REM ** DISTANCIA ENTRE PUNTO Y RECTA **
360 REM *****
370 GOSUB 990
380 LET A=1:GOSUB 750
390 CLS:GOSUB 990:LET A=2:GOSUB 820
400 CLS:GOSUB 990
410 LOCATE 17,27:PRINT "(";X(1);",";Y(1);",";Z(1);")"
420 LOCATE 16,4:PRINT "(";X1(2);",";Y1(2);",";Z1(2);")"
430 LOCATE 12,58:PRINT "(";X2(2);",";Y2(2);",";Z2(2);")"
440 LET V1=X1(2)-X2(2):LET V2=Y1(2)-Y2(2):LET V3=Z1(2)-Z2(2)
450 LET M=SQR((V1^2)+(V2^2)+(V3^2))
460 LET W1=X(1)-X1(2):LET W2=Y(1)-Y1(2):LET W3=Z(1)-Z1(2)
470 LET S1=X(1)-X2(2):LET S2=Y(1)-Y2(2):LET S3=Z(1)-Z2(2)
480 LET P1=W2*S3-W3*S2:LET P2=S1*W3-W1*S3:LET P3=W1*S2-W2*S1
490 LET P=SQR((P1^2)+(P2^2)+(P3^2))
500 LOCATE 12,37:PRINT P/M
510 LINE (150,70)-(500,150)
520 LINE (325,110)-(290,219),,,&H9999
530 GOTO 1070
540 REM *****
550 REM ** DISTANCIA ENTRE DOS RECTAS **
560 REM *****
570 GOSUB 1030
580 LET A=1:GOSUB 820
590 CLS:GOSUB 1030:LET A=2:GOSUB 820
600 CLS:GOSUB 1030
610 LOCATE 6,3:PRINT "(";X1(1);",";Y1(1);",";Z1(1);")"
620 LOCATE 6,58:PRINT "(";X2(1);",";Y2(1);",";Z2(1);")"
630 LOCATE 18,3:PRINT "(";X1(2);",";Y1(2);",";Z1(2);")"
640 LOCATE 14,58:PRINT "(";X2(2);",";Y2(2);",";Z2(2);")"
650 LET V1=X1(2)-X2(2):LET V2=Y1(2)-Y2(2):LET V3=Z1(2)-Z2(2)
660 LET W1=X1(1)-X2(1):LET W2=Y1(1)-Y2(1):LET W3=Z1(1)-Z2(1)
670 LET S1=X1(1)-X1(2):LET S2=Y1(1)-Y1(2):LET S3=Z1(1)-Z1(2)
680 LET P1=W2*V3-W3*V2:LET P2=V1*W3-W1*V3:LET P3=W1*V2-W2*V1
690 LET P=SQR((P1^2)+(P2^2)+(P3^2))
700 LOCATE 10,36:PRINT ABS((S1*P1+S2*P2+S3*P3)/P)
710 LINE (152,90)-(500,90)
720 LINE (152,230)-(500,170)
730 LINE (300,90)-(300,205),,,&H9999
740 GOTO 1070
750 REM *****
760 REM ** SUBROUTINA DE INTRODUCCION DE PUNTO **
770 REM *****
780 LOCATE 7,16:PRINT "COORDENADA X DEL PUNTO ";A;"=";:INPUT X(A)
790 LOCATE 8,16:PRINT "COORDENADA Y DEL PUNTO ";A;"=";:INPUT Y(A)
800 LOCATE 9,16:PRINT "COORDENADA Z DEL PUNTO ";A;"=";:INPUT Z(A)
810 RETURN

```

```

820 REM *****
830 REM ** SUBROUTINA DE INTRODUCCION DE RECTA **
840 REM *****
850 LOCATE 7,16:PRINT "COORDENADA X DE UN PUNTO DE LA RECTA ";A;:INPUT X1(A)
860 LOCATE 8,16:PRINT "COORDENADA Y DE UN PUNTO DE LA RECTA ";A;:INPUT Y1(A)
870 LOCATE 9,16:PRINT "COORDENADA Z DE UN PUNTO DE LA RECTA ";A;:INPUT Z1(A)
880 LOCATE 12,16:PRINT "COORDENADA X DE OTRO PUNTO DE LA RECTA ";A;:INPUT X2(A)
890 LOCATE 13,16:PRINT "COORDENADA Y DE OTRO PUNTO DE LA RECTA ";A;:INPUT Y2(A)
900 LOCATE 14,16:PRINT "COORDENADA Z DE OTRO PUNTO DE LA RECTA ";A;:INPUT Z2(A)
910 RETURN
920 REM *****
930 REM ** CABEDERAS DE PAGINA **
940 REM *****
950 LOCATE 2,16:PRINT "-----"
960 LOCATE 3,16:PRINT "DISTANCIA ENTRE DOS PUNTOS"
970 LOCATE 4,16:PRINT "-----"
980 RETURN
990 LOCATE 2,16:PRINT "-----"
1000 LOCATE 3,16:PRINT "DISTANCIA ENTRE PUNTO Y RECTA"
1010 LOCATE 4,16:PRINT "-----"
1020 RETURN
1030 LOCATE 2,16:PRINT "-----"
1040 LOCATE 3,16:PRINT "DISTANCIA ENTRE DOS RECTAS"
1050 LOCATE 4,16:PRINT "-----"
1060 RETURN
1070 REM *****
1080 REM ** FINAL DEL PROGRAMA **
1090 REM *****
1100 LOCATE 21,16:PRINT "--PULSE UNA TECLA PARA CONTINUAR (S PARA SALIR)--"
1110 LET F$=INKEY$:IF F$="" THEN GOTO 1110
1120 IF F$="S" OR F$="s" THEN CLS:END
1130 GOTO 50

```

Modificaciones para otros equipos

El programa es válido para IBM PC y compatibles. Las modificaciones para el resto de ordenadores son las siguientes:

SPECTRUM

Eliminar línea 10.

Poner en lugar de LOCATE X,Y:PRINT..., la sentencia PRINT AT X,Y;...

Las sentencias LINE (X1,Y1)-(X2,Y2) habrá que realizarlas con PLOT (X1,Y1):DRAW(X2,Y2).

1120 IF F\$="S" OR F\$="s" THEN CLS:GOTO 9999.

AMSTRAD

10 MODE 1.

Sustituir LOCATE X,Y por LOCATE Y,X.

Sustituir las sentencias LINE (X1,Y1)-(X2,Y2) por PLOT (X1,Y1):DRAW (X2,Y2).

COMMODORE

Eliminar línea 10.

Sustituir CLS por PRINT CHR\$(147) en líneas 50, 150, 240, 250, 390, 400, 590, 600, 1120.

Sustituir todas las sentencias gráficas (LINE,LOCATE...) por la subrutina de gráficos para COMMODORE del tomo 1.

MSX

Sustituir LOCATE X,Y por LOCATE Y,X.

Sustituir las sentencias LINE (X1,Y1)-(X2,Y2) por PSET (X1,Y1):DRAW (X2,Y2).

NOTA: El número hexadecimal que aparece al final de alguna sentencia LINE tiene como finalidad que la línea a dibujar sea a trazos discontinuos.

SOCIEDAD

Tipos de lenguajes

Es muy curioso poder hacer una distinción de lenguajes cuando se está trabajando siempre en la misma lengua, y es que, aunque parezca extraño, en una misma lengua, además de muchos dialectos y derivados, hay muchas más formas de expresarse, bien sea por escrito o bien oralmente.

Todos podemos diferenciar claramente el acento regional de una persona al oírle hablar.

También, si nos fijamos en un texto, y en su estructura, en la sintaxis y, hasta en el significado semántico, podemos saber si es un texto periodístico, o es un ensayo, una novela, o un texto científico.

Ya no son sólo los tres géneros literarios principales y sus subgéneros los que podemos diferenciar claramente. El teatro, la prosa y la lírica son géneros literarios que tienen cada uno sus características propias y que los diferencian entre sí desde hace siglos.

Pero en nuestra lengua actual, además de todos estos géneros, coexisten otros tantos que, aunque en menor escala también, tienen características propias que los diferencian de los demás.

Para ayudarte a distinguirlos con mayor claridad presentamos un programa en el que se ofrece una relación de los principales lenguajes que distinguimos actualmente y diez características de cada una de ellos, que te

ayudarán a identificarlos y compararlos tanto semántica como sintácticamente.

Al principio se nos ofrece un menú donde podemos escoger uno de los lenguajes que a continuación se indican y ver sus características: lenguaje coloquial, proverbial, jurídico y administrativo, periodístico y publicitario, técnico y científico.

El programa consta fundamentalmente de una pequeña base de datos donde están debidamente ordenadas las características de los lenguajes referidos anteriormente. Esta "base de datos" está indexada por una matriz, de manera que presenten en pantalla los datos más relevantes de cada lenguaje según los vaya requiriendo el usuario.

```

10 REM *****
20 REM ** TIPOS DE LENGUAJES **
30 REM *****
40 DIM D$(6,10):DIM T$(6)
50 FOR I=1 TO 6
60 FOR J=1 TO 10
70 READ D$(I,J)
80 NEXT J
90 NEXT I
100 FOR I=1 TO 6
110 READ T$(I)
120 NEXT I
130 CLS
140 REM *****
150 REM ** MENU **
160 REM *****
170 PRINT "*****
*****"
180 PRINT "***** T I P O S D E L E N G U A J E S
*****"
190 PRINT "*****
*****"
200 LOCATE 7,24:PRINT "*****"
210 LOCATE 10,24:PRINT "*1 LENGUAJE COLOQUIAL *"
220 LOCATE 11,24:PRINT "*2 LENGUAJE PROVERBIAL *"
230 LOCATE 12,24:PRINT "*3 LENGUAJE JURIDICO Y ADMINISTRATIVO *"
240 LOCATE 13,24:PRINT "*4 LENGUAJE HISPANOAMERICANO *"
250 LOCATE 14,24:PRINT "*5 LENGUAJE CIENTIFICO Y TECNICO *"
260 LOCATE 15,24:PRINT "*6 LENGUAJE PERIODISTICO Y PUBLICITARIO*"
270 LOCATE 16,24:PRINT "*****"
280 LOCATE 20,27:PRINT "---- PULSE LA OPCION ELEGIDA ----"
290 LET A$=INKEY$:IF A$="" THEN GOTO 290
300 IF VAL(A$)>6 OR VAL(A$)<=0 THEN GOTO 290
310 REM *****
320 REM ** BUCLE DE ESCRITURA DE LENGUAJE ELEGIDO **
330 REM *****
340 CLS
350 GOSUB 480
360 FOR I=1 TO 10
370 PRINT I;"-";D$(VAL(A$),I)
380 NEXT I
390 PRINT:PRINT
400 PRINT " ***** PULSE UNA TECLA PARA CONTINUAR *****"
410 PRINT " ***** (F PARA FINAL) *****"
420 REM *****
430 REM ** FINAL DEL PROGRAMA **
440 REM *****
450 LET F$=INKEY$:IF F$="" THEN GOTO 450
460 IF F$="F" OR F$="f" THEN END
470 GOTO 130
480 PRINT "----- LENGUAJE ";T$(VAL(A$));" -----"
490 FOR Y=1 TO 5:PRINT:NEXT Y
500 RETURN

```

APRENDER CON EL ORDENADOR

```
510 REM *****
520 REM ** DATOS **
530 REM *****
540 DATA "SE REFLEJA LA SUBJETIVIDAD DEL HABLANTE", "SE TRATAN SITUACIONES REALES", "SE TRATAN TEMAS GENERALES COMO DUDAS, REPROCHES, NEGACIONES...", "APARECEN INTERJECCIONES", "FIGURAS RETORICAS COMO HIPERBOLES, METAFORAS, INTERROGACIONES RETORICAS"
550 DATA "ECONOMIA DE MEDIOS LINGÜISTICOS", "SE ACOMPAÑA DE GESTOS Y ADEMANES", "USO DE PALABRAS Y ORACIONES INACABADAS", "AUSENCIA DE ADJETIVOS Y ADVERBIOS", "UTILIZACION DE PALABRAS ILATIVAS Y DE 'COMODINES' Y 'MULETILLAS'"
560 DATA "SE INTRODUCEN EN LA CONVERSACION CON ENTONACION AUTONOMA", "SON ORACIONES BREVES (REFRANES)", "NO DEBEN INTERRUPTIR MUCHO EL DISCURSO", "A VECES NO SE ENUNCIAN LOS REFRANES ENTEROS", "SE CONSTRUYEN CON ARTIFICIOS QUE LOS SEPARAN DE LA NORMA GENERAL"
570 DATA "SUELEN COMPONERSE DE DOS PROPOSICIONES", "SUELEN RIMAR EN ASONANTE O CONSONANTE", "ALGUNOS TIENEN ARCAISMOS POR SU PERMANENCIA EN EL TIEMPO", "TAMBIEN REGIONALISMOS Y EXTRANJERISMOS PARA LLAMAR LA ATENCION"
580 DATA "EXISTEN TAMBIEN LAS LLAMADAS 'FORMULAS RIMADAS'", "LENGUAJE PRECISO Y UNIVOCO AUNQUE REITERATIVO", "USO FRECUENTE DEL MODO SUBJUNTIVO", "UTILIZACION DE SINONIMOS PARA EXPRESAR Matices", "EMPLEO DE FORMULAS Y EXPRESIONES FIJAS"
590 DATA "EMPLEO DE EXPRESIONES LATINAS", "USO DE ARCAISMOS", "LENGUAJE CONVENCIONAL EN FORMAS ESTEREOTIPADAS, TRATAMIENTOS O FRASEOLOGIA", "USO DE ORACIONES PASIVAS REFLEJAS E IMPERSONALES", "USO DE PERIFRASIS VERBALES"
600 DATA "UTILIZACION DE LA TERCERA PERSONA IMPERSONAL", "EXTENSION DEL PLURAL", "ADVERBIALIZACION DEL ADJETIVO", "ANTEPOSICION DEL POSESIVO", "EMPLEO DEL 'YO' CON PREPOSICION", "PREFERENCIA DEL PRET. INDEFINIDO ANTE EL PRET. PERFECTO"
610 DATA "VOSEO ENTRE IGUALES Y PARA PERSONAS JERARQUICAMENTE INFERIORES", "DIFERENCIAS LEXICAS EN EL SIGNIFICADO DEL VOCABULARIO", "DIFERENCIAS SINTACTICAS EN LAS CONSTRUCCIONES DE ORACIONES", "DIFERENCIAS FONETICAS COMO SESEO, YEÍSMO..."
620 DATA "UTILIZACION DEL 'USTEDES' EN VEZ DE 'VOSOTROS'", "CREACION DE NUEVOS TERMINOS", "UTILIZACION DE RAICES Y TERMINACIONES LATINAS Y GRIEGAS", "ADMISION DE EXTRANJERISMOS", "UTILIZACION DE TECNICISMOS", "PALABRAS QUE CARECEN DE AMBIGÜEDAD"
630 DATA "ADAPTACION RAPIDA DE AVANCES TECNOLOGICOS AL LENGUAJE", "SEMANTICAMENTE ES UNIVOCO, DENOTATIVO Y PRECISO", "SINTACTICAMENTE SON ORACIONES SIMPLES CON MUCHOS COMPLEMENTOS CIRCUNSTANCIALES", "USO DE ORACIONES ADVERBIALES CONSECUTIVAS CASUALES"
640 DATA "VERBOS EN MODO INDICATIVO QUE DENOTAN OBJETIVIDAD", "FINALIDAD INFORMATIVA", "EN OCASIONES, FINALIDAD PROPAGANDISTICA", "INTENTAN CREAR UNA OPINION EN EL RECEPTOR", "DIVISION DE LAS NOTICIAS EN GRADOS DE IMPORTANCIA"
650 DATA "USO DE 'SLOGANS' EN LOS TEXTOS PUBLICITARIOS", "IMAGENES ICONICAS", "FIGURAS RETORICAS EN LOS ANUNCIOS", "IMPORTANCIA DE LA SITUACION DEL MENSAJE", "UTILIZACION DE LAS NOTICIAS-COMENTARIO Y EDITORIALES"
660 DATA "USO DE MEDIOS DETERMINADOS COMO TELEVISION, RADIO, PRENSA"
670 DATA "COLOQUIAL", "PROVERBIAL", "JURIDICO Y ADMINISTRATIVO", "HISPANOAMERICANO", "CIENTIFICO Y TECNICO", "PERIODISTICO Y PUBLICITARIO"
```

■ Modificaciones para otros equipos

El programa es válido para todos los ordenadores, exceptuando las pequeñas modificaciones que a continuación se refieren:

SPECTRUM

Sustituir todas las sentencias LOCATE X,Y:PRINT... por PRINT AT X,Y:...

```
460 IF F$="F" OR F$="f" THEN GOTO 9999.
```

Las matrices alfanuméricas necesitan una dimensión adicional.

AMSTRAD Y MSX

Sustituir LOCATE X,Y por LOCATE Y,X.

COMMODORE

Sustituir CLS por PRINT CHR\$(147) en líneas 130,340.

Para las sentencias LOCATE, ver simulación de gráficos del tomo 1 de esta obra.

PARA LOS MAS JOVENES

■ Matrículas españolas

Cuando vemos las matrículas de los coches, es curioso el código de letras que se sigue para nombrar el lugar al que pertenece ese coche.

Muchas veces por la carretera puede llegar a ser un problema adivinar el lugar a que se refieren las letras de la matrícula. Este problema no existiría si en las matrículas estuviera escrito el nombre entero de la provincia a la que pertenece; pero eso sería complicado y, además, no cabría la numeración adecuada para identificar el automóvil: imagínate la matrícula de Santa Cruz de Tenerife escrita de esta manera.

Pues bien, como habrás visto, las matrículas, aquí en España, tienen en su parte izquierda una o dos letras mayúsculas, que nos indican a qué provincia pertenecen.

En otros países el código de la población a la que pertenece el vehículo es numérico (así ocurre en Francia).

Este asunto no tendría más complicación si se pudieran diferenciar bien los nombres de las provincias que empiezan por la misma inicial, y que, incluso, algunas continúan con las mismas letras. Entonces, ¿cómo sabemos si LE es de León o Lérica, si V es de Valladolid o de Valencia, si ZA es de Zamora o Zaragoza?

El criterio que se sigue en este caso al matricular un vehículo es asignar menor número de letras (es decir, una) a la provincia

que tiene un mayor volumen de ventas de vehículos. Así, pues, Barcelona ostenta la B, frente a Badajoz, que se representa con BA.

Como este criterio no es bien conocido por todo el mundo, con este programa podrás saber, en cada momento, la provincia a la que pertenece cada símbolo de las matrículas, sin tener que conocer para ello la industrialización de la provincia o su cantidad de habitantes.

En el programa van apareciendo sucesivamente matrículas de diferentes poblaciones, a las que deberás ir respondiendo con la población a la que pertenecen. Cuando ya no falles, puedes pulsar S para salir.

```

10 SCREEN 2
20 REM *****
30 REM ** MATRICULAS DE ESPAÑA **
40 REM *****
50 DIM M$(52):DIM P$(52)
60 FOR I=1 TO 52
70 READ M$(I)
80 NEXT I
90 FOR I=1 TO 52
100 READ P$(I)
110 NEXT I
120 RANDOMIZE TIMER
130 CLS:LET A=0
140 PRINT "***** MATRICULAS DE ESPAÑA *****"
*****
160 FOR I=1 TO 5
170 GOSUB 380
180 LET Q=INT(RND*52)+1
190 LOCATE 5+2*I,10:PRINT M$(Q);"-";
200 FOR J=1 TO 6
210 PRINT INT(RND*9);
220 NEXT J
230 LOCATE 5+2*I,40:INPUT A$:IF A$=P$(Q) THEN LOCATE 5+2*I,69:PRINT ";CORRECTO!"
:GOTO 250
240 LOCATE 5+2*I,69:PRINT ";ERROR!"
250 NEXT I
260 LOCATE 20,10:PRINT "--- PULSE UNA TECLA PARA CONTINUAR (S PARA SALIR) ---"
270 LET X$=INKEY$:IF X$="" THEN GOTO 270
280 IF X$="S" OR X$="s" THEN END
290 GOTO 120
300 DATA "M","T","CR","CU","GU","C","LU","OR","PD","O","VI","BI","SS","NA","B",
"IT","L","GE","U","CS","AL","MU","AB","J","CO","SE","H","CA","MA","GR","CE","ML",
"CC","BA"
310 DATA "A","S","SQ","LO","BU","LE","ZA","SA","VA","PA","PM","AV","SG","TF","GC",
"Z","HU","TE"
320 DATA "MADRID","TOLEDO","CIUDAD REAL","CUENCA","GUADALAJARA","LA CORUNA","LUG",
"O","ORENSE","PONTEVEDRA","OVIEDO","VITORIA","BILBAO","SAN SEBASTIAN","NAVARRA","",
"BARCELONA","TARRAGONA","LERIDA","GERONA","VALENCIA","CASTELLON","ALMERIA"
330 DATA "MURCIA","ALBACETE","JAEN","CORDOBA","SEVILLA","HUELVA","CADIZ","MALAGA",
"GRANADA","CEUTA","MELILLA","CACERES","BADAJOZ","ALICANTE","SANTANDER","SORIA",
"LOGRONO","BURGOS","LEON","ZAMORA","SALAMANCA","VALLADOLID","PALENCIA"
340 DATA "PALMA DE MALLORCA","AVILA","SEGOVIA","TENERIFE","LAS PALMAS","ZARAGOZA",
"HUESCA","TERUEL"
350 REM *****
360 REM ** DIBUJO DE MATRICULA **
370 REM *****
380 LINE (65,80+28*A)-(285,98+28*A),,B
390 LET A=A+1
400 RETURN

```

■ Modificaciones para otros equipos

El programa es válido para el IBM PC. Las modificaciones para el resto de equipos se detallan a continuación:

SPECTRUM

Eliminar línea 10.
50 DIM M\$(52,2):DIM P\$(52,20).
120 RANDOMIZE 0.

APRENDER CON EL ORDENADOR

Sustituir en líneas 190, 230, 240, 260, LOCATE X,Y:PRINT ... por PRINT AT X,Y;...

280 IF X\$="S" OR X\$="s" THEN GOTO 9999.

AMSTRAD

10 MODE 1.

120 RANDOMIZE TIME.

Sustituir en líneas 190, 230, 240, 260 LOCATE X,Y por LOCATE Y,X.

COMMODORE

Eliminar línea 10.

130 PRINT CHR\$(147).

180 LET Q=INT(RND(1)*52)+1.

210 PRINT INT(RND(1)*9).

MSX

10 SCREEN 1.

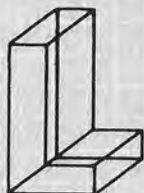
180 LET Q=INT(RND(1)*52)+1.

210 PRINT INT(RND(1)*9).

Eliminar línea 120.

PEQUEÑA HISTORIA DE LA INFORMATICA

La segunda generación de ordenadores



A segunda generación de ordenadores está marcada por la utilización del transistor, como elemento importantísimo de los circuitos de los procesadores centrales. El paso de las válvulas de vacío a los transistores es fundamental, ya que se producen cambios notables en las máquinas. En primer lugar, son mucho más fiables, ya que los transistores lo son, frente a los tubos de vacío. En segundo lugar, y relacionado con lo anterior, el consumo de potencia se reduce enormemente. Y ya en último lugar, y también importante, los nuevos ordenadores tienen un tamaño sensiblemente menor que sus predecesores. En suma, se trata de una tecnología muy diferente de la anterior, que supondrá un avance en el que basarse para los nuevos descubrimientos que revolucionarán la informática.

No vamos a hablar del descubrimiento del transistor, ya que lo hemos hecho en un recuadro aparte. Recomendamos, pues, al lector que lea dicho recuadro antes de proseguir con estas líneas.

El primer ordenador fabricado con esta nueva tecnología fue el SEAC. Este ordenador había sido pensado para instalarlo en un Laboratorio Meteorológico, y estuvo funcionando desde la mitad de nuestro siglo: 1950. No estaba, sin embargo, totalmente transistorizado, ya que además de los 10.000 diodos, todos ellos de germanio, disponía de 750 válvulas de vacío. Este ordenador no dio un buen resultado. Las conexiones no eran todo lo perfectas que cabía esperar, y como resultado se producían demasiados fallos, que hicieron al ordenador menos fiable de lo que especificaban sus creadores.

¿Sabía usted que...

Desde el año 1930 se estaban realizando trabajos con distintos tipos de cristales, estudiando su conductividad. Algunos elementos se podían considerar claramente como «conductores», ya que eran capaces de conducir la electricidad en cualquier dirección. Sin embargo, existían otros elementos de propiedades eléctricas intermedias entre los elementos conductores de cobre, oro, plata, etc., los elementos claramente no conductores, como, por ejemplo, el cuarzo. Entre los elementos intermedios se encuentran el selenio y el germanio. Estudiando con más dedicación estos elementos que apenas si habían tenido interés práctico alguno hasta entonces, la compañía americana Bell comenzó a considerar sus posibilidades, llegando a crear un laboratorio especial para continuar estudios y pruebas. Hacia el año 1945 se crearon en dicho laboratorio unos amplificadores fabricados, utilizando los elementos con los que estaban experimentando: selenio y germanio. Los investigadores de Bell Lab observaron que introduciendo impurezas dentro de los cristales de selenio o germanio podían crearse distintas zonas con características diferenciadas dentro del material: unas podían tener un exceso de electrones (y, por tanto, podían actuar como emisores o colectores) y otras zonas un déficit (y, por tanto, podrían utilizarse como base). Para amplificar la señal se modificaría la polarización, de la unión base-emisor, variando el nivel de escasez o deficiencia de electrones.

El transistor es obra de tres grandes científicos del laboratorio de la Bell: J. Bardeen, Brattain y W. Shockley. El primer tipo que crearon era de germanio, y tenía unos contactos metálicos muy finos (que serían causantes de los primeros problemas, resueltos satisfactoriamente poco tiempo después). Este transistor fue concluido en enero de 1947. El transistor supuso una verdadera revolución tecnológica. Su principal virtud frente a los tubos era el ser

PEQUEÑA HISTORIA DE LA INFORMATICA

mucho más robusto. Pero, además, era mucho más pequeño, y sobre todo consumía mucha menos potencia. Los equipos de transistores eran mucho más fiables que los que utilizaban tecnología a base de tubos. En estos últimos los cátodos funcionaban siempre a temperaturas muy altas, por lo que solían causar la mayor parte de las averías. El bajo consumo fue también un factor muy importante, no sólo por el ahorro de potencia que suponía, sino también por los problemas que eliminaba, al disminuir muchísimo la disipación de calor, fuente casi constante de problemas.

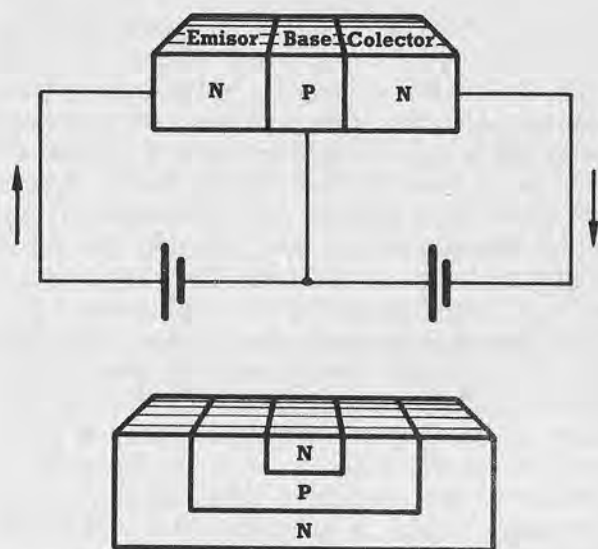
La tecnología de fabricación de los transistores ha ido cambiando con el tiempo. Los primeros transistores eran de germanio principalmente. En los años sesenta fue prosperando la llamada «tecnología plana», que formaba el transistor con tres regiones o zonas planas, que eran inyectadas dentro de la «tableta» de silicio. Se introducían estas impurezas (de distinto tipo, según fuera su misión dentro del transistor), consiguiendo, por ejemplo, los contactos eléctricos mediante pequeñísimos depósitos de aluminio dentro de la propia oblea, a la que también se le inyectaba fósforo, etc., para crear el transistor dentro de la misma oblea, siendo esta idea fundamental la base de lo que más tarde serían los circuitos integrados y la miniaturización.

El siguiente intento se diseñó en 1954, y la empresa responsable fue la Philco Corporation. Esta empresa había experimentado bastante con numerosos tipos de transmisores, obteniendo una clase llamada «de barrera de superficie», que sería la base de la mayoría de los transistores utilizados en la segunda generación de ordenadores. La empresa fue muy activa y consiguió varios contratos con organismos norteamericanos, entre ellos uno con la National Security Agency (organismo cuyo cometido es en cierto modo «paralelo» al de la CIA, pero mucho menos conocido, y, por ende, menos controlado a todos los niveles y del que ya hemos contado algún pequeño dato curioso), y también con la Marina norteamericana. Para la National Security Agency, la Philco diseñó el TRANSAC S-1000, utilizando los transistores con los que había estado investigando. Del resultado de este ordenador se sabe menos que sobre otros ordenadores instalados en organismos «algo más comunicativos». Para la Marina norteamericana creó el CXPQ, otro ordenador también totalmente transistorizado. El resultado de este último ordenador fue bastante bueno, y decidió a la Philco a realizar un

nuevo diseño para otro ordenador basándose principalmente en el CXPQ de la Marina, aunque, naturalmente, con innovaciones. La nueva máquina fue la TRANSAC 2000. El proyecto de este ordenador se había basado bastante en estudios y pequeños proyectos del Institute of Advanced Studies. Era muy buena máquina porque disponía además (aparte del procesador central), de otros órganos muy avanzados, entre los que cabe citar el sistema de cintas magnéticas, a la cabeza de los restantes sistemas equivalentes. El equipo tuvo bastante éxito comercial. Naturalmente, no es lo mismo tener éxito en la comercialización de un ordenador de los años sesenta, pongamos por caso, que en los cincuenta, o que hoy en día. Los primeros ordenadores que se fabricaron se habían creado prácticamente «a medida» del organismo que los iba a recibir. Poco a poco, se fue ampliando el abanico de organismos, empresas que compraban ordenadores, pero seguía siendo un producto especialísimo, muy caro, sólo asequible a empresas muy poderosas, y por tanto casi reducido a las necesidades de organismos estatales o paraestatales.

El TRANSAC 2000 no pudo salir al mercado hasta 1960, algo después del anuncio, por parte de IBM de su nueva serie 7090. Ante la fuerte competencia, la Philco puso el máximo interés en obtener transistores de mejores características, intentando especialmente obtener velocidades superiores para el procesador central. El producto resultado de todos estos esfuerzos fue el Philco 212, con calidad suficiente para competir con el 7094 de IBM y con el CDC 3600 de Control Data Corporation. Sin embargo, la compañía Philco tenía problemas de financiación, que fue arrastrando durante largo tiempo. Llegó a un punto en el que le fue imposible continuar y fue absorbida por la compañía de automóviles Ford. Al cabo de un corto período de tiempo, la Ford abandonó esta rama, y la compañía desapareció.

Las grandes casas fabricantes de ordenadores seguían trabajando, intentando mejorar a la competencia, que cada vez era más numerosa. Además, en esos años (e incluso hasta nuestros días) ha sido relativamente frecuente que se crearan otras compañías de ordenadores a partir de pequeños grupos de técnicos e ingenieros que trabajaban en grandes compañías y veían el provecho que podían obtener de sus ideas, si conseguían la adecuada financiación. Así sucedió con la Control Data Corporation, creada por un gru-



po de ingenieros de la poderosa Remington Rand. Antes de crear esta compañía habían dado vida a otra, ERA, de la que salieron algunos ordenadores exitosos. El presidente de la Control Data era William C. Norris, muy conocido como los restantes miembros del grupo, entre los que estaba Seymour Cray. El 1604 de CDC fue diseñado casi totalmente por Cray (que luego formaría su propia compañía para diseñar los enormes y potentísimos ordenadores Cray). El CDC 1604 era una máquina muy completa. Contenía unos 25.000 transistores, más de 5.000 diodos y una memoria de ferrita muy amplia (32.768 palabras de 48 bits). Este ordenador abriría el camino que tomó CDC en sus máquinas: generalmente eran ordenadores muy grandes y potentes, especialmente pensados para resolver problemas científicos. Pero expliquemos un poco las técnicas comerciales que utilizó el CDC y que le llevaron al éxito, a pesar de las muchas trabas que las grandes compañías como IBM y Remington Rand le iban poniendo en su camino. Control Data volvió los ojos de nuevo a las universidades. Pensó que tenían un potencial humano muy interesante, el cual, si trabajaba con sus máquinas, se acostumbraría a ellas pasando a ser clientes en el futuro. CDC vendía máquinas a las universidades a precios muy bajos, y prácticamente sin software. Este último punto para cualquier universidad más que ser un problema era un desafío. La necesidad de desarrollar programas interesantes sin recortes por motivos económicos era un reto apetecible. (En una universidad, el tiempo empleado en el desarrollo de cualquier proyecto no tiene tantas cortapisas económicas como en una empresa.)

Otro ordenador con «personalidad» de esta segunda generación de ordenadores fue, sin duda, el Atlas Guidance Computer, máquina muy potente y eficaz cuyo cometido era controlar la puesta en órbita de los satélites de comunicaciones. También se utilizó para el control de misiles intercontinentales, que eran una de las armas más modernas en aquellas fechas. El ordenador se ubicó en Cabo Cañaveral, y funcionó desde 1958 hasta 1961.

Univac e IBM, mientras tanto, iban colocando trabas de patentes a estas pequeñas compañías que se iban formando. No hay que olvidar que disponían de muchos más recursos, pero también se gastaban en investigación mucho dinero. También es cierto que siempre tenían el riesgo de que algún jefe de proyecto les abandonara para crear su propia compañía de ordenadores. Estas grandes compañías disponían de gabinetes jurídicos con un nutrido personal, muy escogido. Estaban en continuo litigio por problemas de patentes, que a veces podían durar años.

El Univac 1103 y el IBM 650 fueron máquinas transistorizadas, aunque no todos los ordenadores de esta segunda generación estaban totalmente transistorizados. Para muchos, la segunda generación de ordenadores comienza en 1959, año en el que varias compañías de ordenadores sacaron al mercado productos totalmente transistorizados. Entre los ordenadores que aparecieron a partir de junio de 1959, por orden cronológico, citaremos los equipos siguientes: GE 210, IBM 1401, IBM 1620, IBM 7090, NCR 304 y RAC 501. De todas estas nuevas casas fabricantes iremos hablando poco a poco.

La segunda generación de ordenadores también se distinguió por una mejora notable de los sistemas de almacenamiento de la información. Las memorias de núcleos de ferrita permitían almacenar un volumen de datos mucho mayor, y además eran mucho más fiables. No hay que olvidar que la capacidad de memoria es una de las características fundamentales en un equipo, tanto que en muchos, este dato aparece en algún modo en la denominación de cada modelo.

Pero, probablemente, las velocidades de cómputo que podían obtenerse gracias a los transistores, y la mayor fiabilidad de las máquinas fuera la característica más evidente a los ojos del nuevo usuario, deseoso de pasar a utilizar estas atractivas máquinas. Los ordenadores de la segunda generación eran efecti-

PEQUEÑA HISTORIA DE LA INFORMATICA

vamente «máquinas atractivas». Ya no ocurría como en la primera generación en que los ordenadores sólo interesaban a círculos pequeños de científicos o técnicos, empeñados en la resolución de un problema concreto, y a un pequeño grupúsculo de personas no relacionadas directamente con esta industria. El ordenador había salido a la calle, interesaba al gran público, era cada día que pasaba algo más accesible.

Efectivamente, entre los ordenadores de la primera y segunda generación existían diferencias notables. El transistor fue el «emblema» de este cambio y su primera razón de ser, pero las nuevas máquinas tenían muchas otras diferencias (sobre todo desde el punto de vista conceptual y funcional). Más que un mero avance tecnológico, el transistor supuso

un nuevo modo de concebir la informática. Los ordenadores se iban creando pensando en facilitarle la labor al usuario. Las máquinas iban siendo cada vez más potentes, adaptándose cada día a nuevas aplicaciones, y ampliando mercados. Pero también comenzaron a aparecer otros ordenadores más pequeños (y, por tanto, más sencillos y más baratos) que resolvían problemas específicos. Esta especialización de las máquinas fue muy útil, porque permitió acceder al grupo de usuarios a muchos empresarios de nivel medio. A este último punto ayudó mucho también el sistema de proceso en batch (por lotes), que permitía aprovechar mucho más la hora de ordenador. Otros pequeños «trucos» para rentabilizar las máquinas fueron la utilización del ordenador en tiempo compartido.

■ Control del proceso deductivo

Aunque, como se ha visto, el proceso de obtención de una conclusión (mediante aplicación de la regla general de deducción, «modus ponens», a un hecho conocido) es un paso sumamente simple, el proceso general deductivo es más complejo. Hay dos problemas básicos en el funcionamiento del motor de inferencia:

a) ante todo, cómo comenzar el proceso y/o cuál es el objetivo a alcanzar: en efecto, la base de conocimientos es un cúmulo de información estática y habrá que arrancar el proceso de algún modo. Pueden utilizarse procedimientos diferentes: en ocasiones se marca(n) alguno(s) hecho(s) básico(s) con los que el motor debe comenzar su actividad deductiva; en otros casos se marca un objetivo a obtener (un hecho a demostrar) y es tarea del sistema ver si es demostrable o no; hay sistemas que admiten la presentación de un objetivo genérico (un conjunto de varios hechos que pueden ser demostrados) y el sistema elabora un proceso deductivo para ver, a partir de los hechos ya conocidos, cuál es el que se puede demostrar... ¡veremos esto en detalle a continuación!

b) por otro lado, el motor de inferencia debe resolver los conflictos que se presentan cuando, en su tarea deductiva, puede elegir varias líneas de razonamiento alternativas (a partir de diferentes reglas que pueden ser aplicables). Debe disponer de «normas» de funcionamiento o «reglas sobre cómo aplicar las reglas»; o bien, debe tener capacidad para consultar al usuario... o ambas cosas a la vez.

Se puede hablar de unas técnicas generales de conducción del proceso deductivo: encadenamiento adelante o atrás, búsqueda en profundidad o en anchura, utilización de reglas de tipo «ver primero» o heurísticas (control por el contenido de las reglas), etc., aunque existen numerosos procedimientos de control adicionales que se suelen utilizar en unos u otros de los sistemas expertos y/o herramientas disponibles (activación y desactivación de paquetes de reglas o control por «objetivos ficticios» y «datos ficticios» en Intelligence Service; «gestión de hipótesis» y utilización de primitivas de metaconocimiento en SNARK; control de reglas que se activan pasado un umbral de certeza y reglas de uso exclusivamente en encadenamiento adelante en M.I.M.I., etc.).

Veremos en detalle las técnicas generales citadas anteriormente y comentaremos los restantes procedimientos de control para que el lector tenga una visión lo más completa posible de las técnicas utilizadas para la gestión del proceso deductivo en los sistemas basados en el conocimiento.

■ Encadenamiento adelante y atrás

Si arrancamos el proceso deductivo a partir de un conjunto de reglas y uno o varios hechos conocidos, se puede aplicar la regla de «modus ponens» para ir obteniendo hechos nuevos demostrados. Así se va «avanzando» a través de la base de conocimientos obteniendo sucesivamente nuevos hechos a utilizar en los pasos posteriores: esto es el encadenamiento hacia adelante.

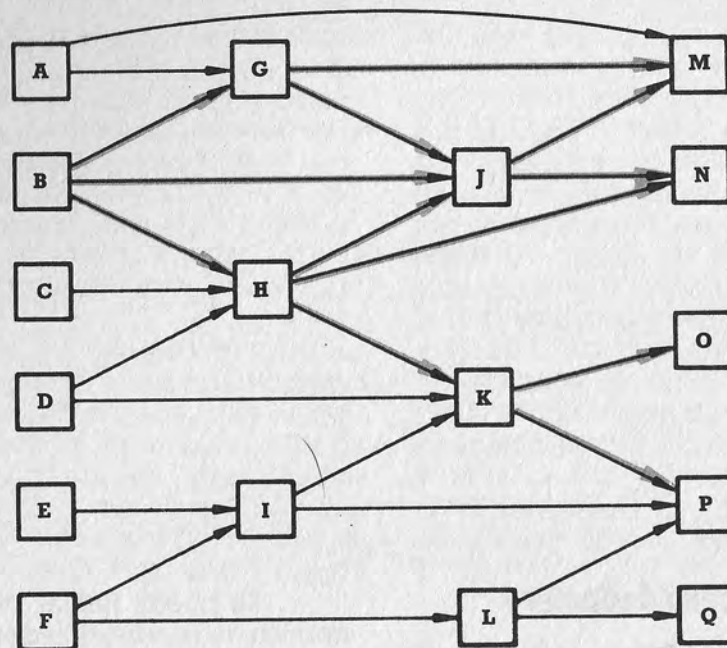


Fig. 1. Proceso de encadenamiento hacia adelante a partir de un hecho (B) avanzando por la base de conocimientos.

Si, por el contrario, queremos llegar a una conclusión (objetivo a obtener), se puede examinar qué reglas de producción tienen en su conclusión ese objetivo. Los antecedentes de la o las reglas que cumplen esa condición son los nuevos objetivos (se llaman a veces subobjetivos) a obtener, pues si podemos demostrar estos subobjetivos, automáticamente la aplicación de la regla correspondiente permitirá obtener el objetivo buscado. Con cada

uno de los subobjetivos se realiza la misma operación buscando las reglas en que son conclusión: los antecedentes (parte de condición o premisas) de esas reglas darán los nuevos subobjetivos a obtener... y por este procedimiento vamos «retrocediendo» por la base de conocimientos hasta los primeros hechos que conocemos o sobre los que se pregunta al usuario. Este es el encadenamiento hacia atrás.

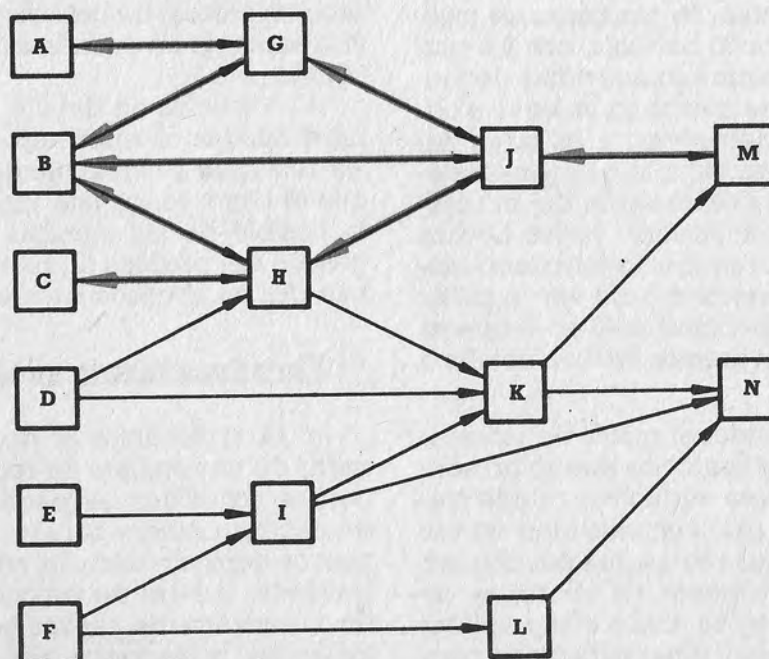


Fig. 2. Proceso de encadenamiento hacia atrás a partir del objetivo M retrocediendo hacia los hechos de base A, B, C,...

■ Búsquedas en profundidad y en anchura

En la búsqueda de nuevos hechos a través de la base de conocimientos podemos «desplazarnos» hacia atrás o hacia adelante,

como hemos visto. Pero, por otro lado, hay que tener en cuenta otra posibilidad. En la búsqueda con encadenamiento hacia atrás se parte de un objetivo para averiguar los hechos de que hay que disponer para poderlo demostrar. Se toma entonces el primero de estos hechos como nuevo subobjetivo y así se llega has-

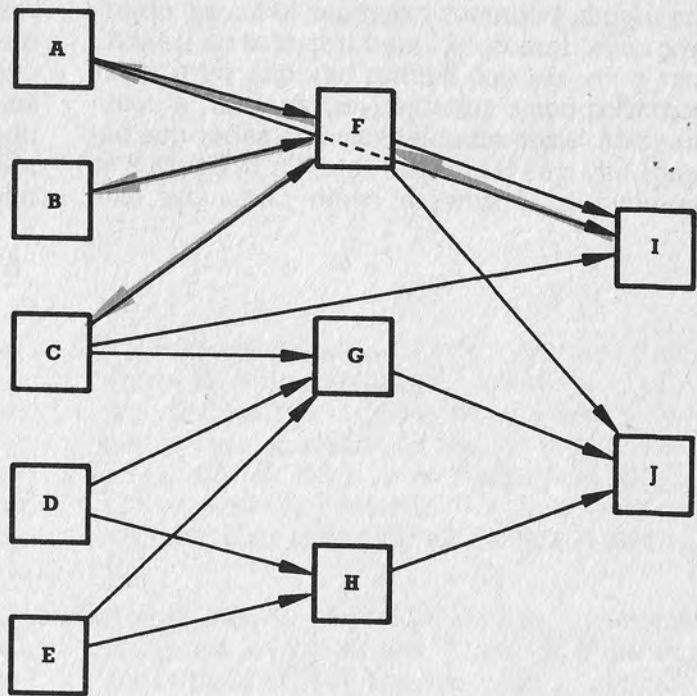


Fig. 3. Proceso de búsqueda en profundidad con en-
cadenamiento hacia atrás.

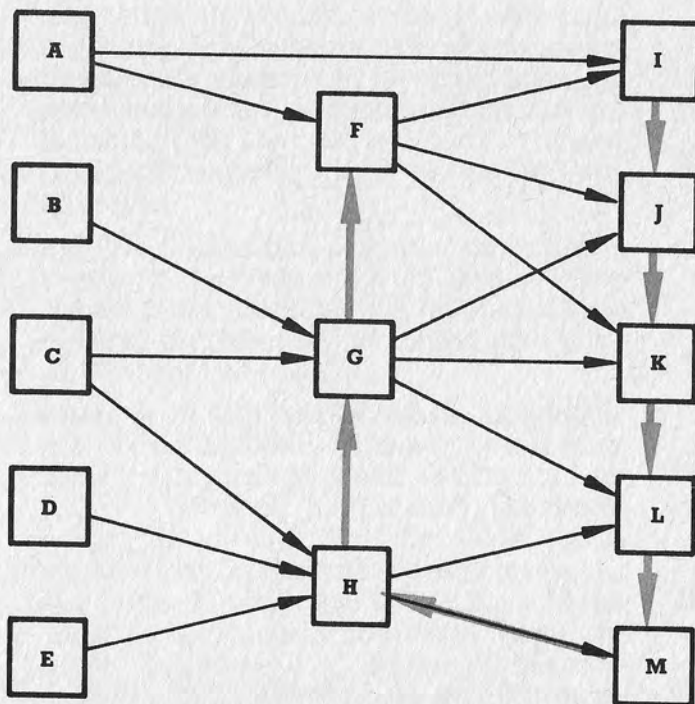


Fig. 4. Proceso de búsqueda en anchura con encadenamiento hacia atrás.

ta los datos de base necesarios (ver fig. 1). Sólo después de llegar hasta un hecho de base se suele realizar el encadenamiento hacia adelante para volver después a considerar el siguiente subobjetivo. Este esquema de análisis del conocimiento se llama búsqueda en profundidad.

Pero no es éste el único camino posible; en efecto, podemos examinar todos los objetivos cuya demostrabilidad tratamos de establecer y ver de qué hechos hay que partir para tomarlos como subobjetivos; analizar, a continuación, estos subobjetivos para saber qué hechos hay que demostrar, etc. En la figura 2 se muestra gráficamente cómo podía ser este

otro camino. A este esquema de análisis del conocimiento se le llama búsqueda en anchura o en amplitud.

Naturalmente, se puede programar un proceso de búsqueda en anchura junto con un esquema de encadenamiento hacia adelante.

En general, se puede decir que si el número de hechos de partida no es muy numerosos, pero los hechos a demostrar sí lo son, una búsqueda hacia adelante será más adecuada; si, por el contrario, se dispone de un solo objetivo (o varios, pocos) y los datos (hechos) de los que se dispone o puede disponer son numerosos, será más eficaz el encadenamiento hacia atrás.

GLOSARIO DE TERMINOS UTILIZADOS EN SISTEMAS EXPERTOS

Dependencia. Relación entre una consecuencia y sus antecedentes, que debe ser almacenada si el proceso de razonamiento tiene que examinarse retrospectivamente. Se llama red de dependencias a un gráfico en el cual los nodos representan aserciones y los arcos u otros nodos representan relaciones.

Discriminación. Proceso de aprendizaje que distingue las instancias de un concepto de las no instancias. Se llama también discriminación al acto de refinar una supergeneralización de un concepto que debe aprenderse para excluir las no instancias que fueron clasificadas erróneamente como instancias del concepto. También puede referirse a un mecanismo de aprendizaje específico que explota la retroalimentación relacionada con las clasificaciones erróneas, para refinar una supergeneralización de un concepto.

Elemento de control. Elemento de la memoria de trabajo cuya única utilidad es almacenar conocimientos de control. Como ejemplo de elemento de control, citaremos un elemento de contexto.

Elemento de una condición. En algunos casos, el lado izquierdo de una regla en un sistema de producción se expresa como conjunto de patrones (o máscaras) que deben contrastarse con el contenido de la memoria de datos. Se denomina elemento de una condición a cada uno de estos patrones. Cuando se instancia una regla, cada elemento de condición debe ser tal que concuerde con un elemento de la memoria de datos.

Elemento resultante. Estructura temporal utilizada en los lenguajes orientados al objeto de construir una descripción de un elemento de la memoria de trabajo con la finalidad de modificar la memoria de trabajo. Las acciones de llamada y modificación se implementan utilizando elementos resultantes.

Especificidad. Estrategia de resolución de conflictos que prefiere instanciaciones de reglas más específicas, medidas por lo general por el número de variables o constantes o números de las comprobaciones del lado izquierdo. Este principio incluye el heurístico de que las reglas con antecedentes más detallados son más discriminatorias que las que lo son menos, produciendo por lo general mejores resultados.

Estrategia de resolución de conflictos. Es un principio específico aplicable para poder ordenar parcialmente las instanciaciones del conjunto conflicto. Aquellas instanciaciones que aparecen dominadas por otras, de acuerdo con este principio, se eliminan del conjunto de conflictos, y se descarta la posibilidad de activarlas en ese ciclo.

Filtro de reglas. Restricción de la parte de memoria de producción que participa en el proceso de concordancia con un subconjunto, en aras de una mayor eficiencia.

Heurístico (regla heurística). Principio o regla elemental que comporta algún conocimiento de resolución de problemas, y que tiene la particularidad de llevar a la solución del problema más rápidamente que el

TERMINOLOGIA

método general de resolución implícito en un determinado algoritmo, reduciendo considerablemente la búsqueda. Sin embargo, no garantiza su funcionamiento en todas las situaciones. Este término fue popularizado principalmente por el matemático G. Polya.

Intérprete. En un sistema experto, aquella parte de la máquina de inferencias que decide cómo aplicar el conocimiento del dominio. En un sistema de programación, aquella parte del sistema que analiza el código para decidir qué acciones deberán llevarse a cabo a continuación.

Lado izquierdo. Una de las dos partes de una regla. La otra es el lado derecho. El lado izquierdo especifica los antecedentes que deben satisfacerse si se aplica la regla. En las reglas de gramática, el lado izquierdo especifica una cadena de símbolos que pueden sustituirse por otra cadena de símbolos.

El sistema de producción que utilizan estrategias de encadenamiento hacia atrás, el lado izquierdo especifica los subobjetivos del objetivo que está especificado en el lado derecho. En los sistemas de producción de encadenamiento hacia adelante, el lado izquierdo consiste en un conjunto de elementos de condición que deben contrastarse con el contenido de la memoria de datos.

Lenguaje orientado a los sistemas de producción. Lenguaje de ordenador que emplea como componente fundamental una arquitectura que es un sistema de producción.

Método de solución de problemas. Procedimiento (algorítmico o heurístico) por el cual se busca la solución a un problema.

Producción. Término utilizado por la filosofía cognoscitiva para describir una regla SI-ENTONCES.

Programa basado en reglas. Programa similar en cierto modo a un programa de sistema de producción en que el conocimiento se representa explícitamente mediante reglas en lugar de hacerlo mediante procedimientos, pero los programas basados en reglas no están implementados necesariamente en los lenguajes de sistemas de producción de propósito general, y no necesitan hacer uso exclusivo de la arquitectura de los sistemas de producción.

Prueba de teoremas por resolución. Es una de las utilizaciones específicas de la lógica deductiva para probar teoremas en el cálculo de predicados de primer orden. El método utiliza el siguiente principio de resolución: $(A \vee B)$ y $(\neg A \vee C)$ implica $(B \vee C)$.

